Commodore
COMPUTER
CLUB
39
L. 4.000
La rivista degli utenti di sistemi Commodore
25 febbraio 1987 - Anno VI - N. 39 - Sped. Abb. Post. Gr. III/70 - CR - Distr. MePe
Inserto: come gestire
i file sequenziali
Grafica: il giardino
elettronico
Tavolozza per C128
Autorun per C64,
C16 e Plus4
Quattro games
da copiare
Esso
Esso
systems

Sommario

INSERTO

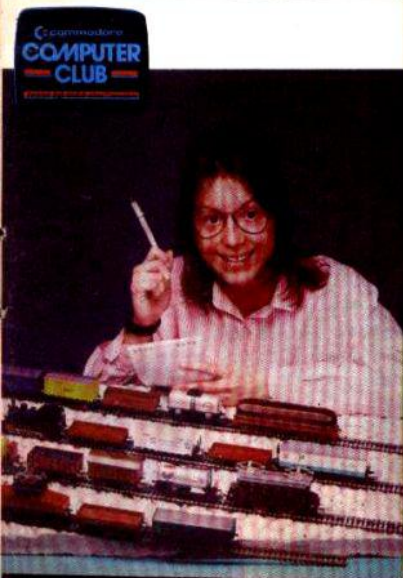
COME GESTIRE
SENZA PROBLEMI
I FILE SEQUENZIALI

RUBRICHE

4 L'ARGOMENTO DEL MESE
5 DOMANDE/RISPOSTE
40 DIRECTORY 5
68 RECENSIONI

PAG.	REMarks	Vic 20	C 64	C16/128	Gen./Amiga
	Giochi				
10	Defender of the crown				•
14	Un ritardo promettente	•	•	•	•
28	Uno scacciapensieri a zero lire		•	•	
32	Un sistema controllato	•	•	•	•
	Intervista				
13	Il futuro dell'utente Commodore				
	Didattica				
18	Peek, Poke & Sys		•	•	
	Grafica				
21	Il giardino elettronico			•	
43	Curve e simmetrie		•	•	
90	Una tavolozza per il tuo C/128			•	
	Protezioni				
36	Autorun Maker		•	•	
38	Autorun per C/16 e Plus/4			•	
	Periferiche				
42	Come fare una tabella	•	•	•	•
	Gw-Basic				
61	I file sequenziali in ambiente Gw-Basic		•		•
	Oltre il Basic				
72	Voltiamo pagina	•	•	•	•
	Enciclopedia di routine				
83	Parlar di sprite	•	•	•	•
	Enciclopedia L.M.				
94	Uno schermo più allegro		•		

Direttore: Alessandro de Simone
Redazione/collaboratori: Claudio Baiocchi, Carlo e Lorenzo Barazzetta, Giovanni Bellù, Simone Bettola, Andrea e Alberto Boriani, Diego e Federico Canetta, Giancarlo Castagna, Umberto Colapicchioni, Pasquale D'Andreti, Maurizio Dell'Abate, Valerio Ferri, Luca Galluzzi, Michele Maggi, Giancarlo Mariani, Marco Miotti, Flavio Molinari, Claudio Mueller, Massimo Pollutri, Carla Rampi, Fabio Sorgato, Giovanni Verrelli, Antonio Visconti.
Segreteria di redazione: Maura Ceccaroli,
Ufficio Grafico: Arturo Ciaglia
Direzione, redazione, pubblicità: V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Milano: Leandro Nencioni (direttore vendite), Giorgio Ruffoni, Claudio Tidone - V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
- Emilia Romagna: Spazio E - P.zza Roosevelt, 4 - 40123 Bologna - Tel. 051/236979
- Toscana, Marche, Umbria: Mercurio Srl - via Rodari, 9 - San Giovanni Valdarno (Ar) - Tel. 055/947444
- Lazio, Campania: Spazio Nuovo - via P. Foscari 70 - 00139 Roma - Tel. 06/8109679

Segretaria: Marina Vantini - **Abbonamenti:** Paola Bertolotti
Tariffe: prezzo per copia L. 4.000. Abbonamento annuo (11 fascicoli) L. 40.000. Estero: il doppio.
- Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 80.000
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
Composizioni: Systems Editoriale Srl - **Fotolito:** Systems Editoriale Srl
Stampa: La Litografica S.r.l. - Busto Arsizio (VA)
Registrazione: Tribunale di Milano n. 370 del 2/10/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70% - **Distrib:** MePe, via G. Carcano 32 - Milano

Il mondo è dei dati

Sapete che siamo circondati, senza saperlo, da milioni di bit che ci riguardano da vicino?

La vostra carta di identità contiene dati memorizzati nel cervello elettronico sistemato nel municipio della vostra città.

Questo può essere interrogato, in qualsiasi momento, da un cervellone più grande di lui manovrato, si spera, da personale autorizzato a farlo (dipendenti del ministero dell'Interno, funzionari di Polizia, Carabinieri eccetera).

I dati relativi alla vostra automobile o ciclomotore, inoltre, sono su più supporti magnetici: su quello dell'Automobile Club, dell'assicurazione, della fabbrica in cui è stata prodotta, del concessionario dal quale è stata acquistata; e chissà dove altro ancora.

Le date delle vostre vaccinazioni, utili a voi personalmente, sono anche a disposizione delle autorità sanitarie che, magari, le scambiano con quelle di Paesi stranieri per effettuare statistiche di ogni tipo.

La stessa sorte seguono i vostri dati relativi ad atti burocratici (compravendite, contratti, INPS, codice fiscale, libretto di banca ed altro) e anche, ovviamente, i miei e quelli di tutti gli altri cittadini del cosiddetto mondo civilizzto.

Fa un certo effetto pensare che, mentre state leggendo il giornale, il vostro nominativo viene scartato nella ricerca di quello il cui titolare ha richiesto un certificato di nascita oppure, ve lo auguriamo, nell'individuazione di chi ha evaso le tasse.

In questo momento, forse, le macchine automatiche delle Poste stanno smistando la corrispondenza a voi destinata oppure proprio nella sede da cui scrivo, la nostra simpatica Paola, due porte più in là, sta inserendo il vostro nominativo nel file dei nuovi abbonati.

File, dati, bit, byte che corrono lungo i cavi, accompagnati, o meno, da bit di controllo, di parità, di impostazione di velocità fino a capitare in un'altra Ram oppure depositati su supporto magnetico o, ancora, decodificati e scaricati su stampante; oppure inviati altrove, per chissà quale ulteriore manipolazione o controllo.

Un turbinio silenzioso e inavvertibile di bit da qualche parte fa girare il motore di un drive, avanzare un nastro, spostare una margherita; un pixel si illumina mentre un altro si spegne o, addirittura, un sottile raggio laser è incaricato di scrivere, o di leggere dati memorizzati in precedenza.

E il nostro home computer, anche lui, è in grado certamente di gestire una certa quantità di dati. Gli stessi programmi non sono altro che una successione ordinata di byte da decodificare e ai quali obbedire senza esitazione.

E anche le Rom, a pensarci bene, sono anch'esse dei file, e così anche, perchè no?, la tastiera...

Non potevamo quindi fare a meno di affrontare uno dei temi vitali per un concreto utilizzo di un computer.

L'inserto sulla gestione dei file, su come usarli e, soprattutto, su come NON usarli è già stato affrontato in precednza, ma non in maniera sistematica.

Ma quali file gestire con un modesto Commodore 64? Non certo un'agenda telefonica, noiosa ma didatticamente indispensabile; nè una procedura di fatturazione e gestione di magazzino, per i quali è necessario un computer più potente e più veloce.

Rimane, comunque, una gran quantità di dati che, elaborati da un home computer, possono esser memorizzati per poi riprenderli e manipolarli ulteriormente: file di word processor, di spreadsheet, di PICCOLI data base. E, ancora, file generati dall'elaborazione di sistemi di equazioni, dallo sviluppo di schedine Totocalcio oppure di archivio del Lotto.

La memorizzazione dei dati risultanti da un'inchiesta potranno, in seguito, esser riletti un'infinità di volte e sottoposti alle statistiche più fantasiose.

In questo numero, quindi, troverete materiale per informarsi su un argomento, la gestione dei file, che più di ogni altro giustifica la nascita e, per ciò che ci riguarda, la diffusione del personal computer.

Alessandro de Simone

Musica & computer

□ **Avendo letto il N.33 di Commodore Computer Club (dedicato alla musica), vorrei avere ulteriori notizie sulla possibilità di utilizzare il computer in campo musicale e, se possibile, in campo editoriale per facilitare la scrittura di musica mediante programmi del tipo "Music Shop".**

(Eugenio del Sarto - Alessandria, Claudio Trinoli - Castelnuovo di Assisi)

• Grazie ad una particolare interfaccia, chiamata MIDI, il Commodore 64, e altri computer, consentono, tra l'altro, di sincronizzare i ritmi di strumenti musicali elettronici collegati tra loro.

Sono disponibili numerosi modelli di tastiere, programmi e apparecchi vari che, facendo capo ad uno o più computer, costituiscono una vera e propria orchestra gestibile da calcolatore.

E' piuttosto difficile, come potrai intuire, dare consigli sul modello più opportuno o sul software più adeguato alle varie esigenze. Se, però, ti rivolgi ai componenti di complessi musicali, riuscirai sicuramente ad avere notizie "dirette" su come risolvere i tuoi problemi: nessuno potrà risponderti più esaurientemente di coloro che utilizzano abitualmente strumenti musicali collegati con il computer.

Per ciò che riguarda la seconda parte della domanda, vale a dire la possibilità di adoperare il C/64 per scrivere musica, è probabile che la qualità di stampa del Music Shop, benchè ottima per usi... domestici, non sia adeguata per realizzare spartiti musicali da riprodurre per scopi editoriali: anche l'occhio vuole la sua parte e la definizione ottenibile con una semplice stampante ad aghi non penso che possa competere con la qualità riscontrabile negli spartiti posti in vendita nei negozi di musica.

A parte la velocità massima con cui puoi scrivere musica utilizzando programmi "musicali", è infatti presente il problema della qualità di stampa legata inevitabilmente ai limiti della stessa printer.

Una qualità migliore si può sicuramente ottenere con stampanti a getto di inchiostro oppure con le moderne Laser Printer, dal costo relativamente elevato, da collegare, però, a computer professionali del tipo Amiga, Macintosh o Ms-Dos.

Se ritieni che valga la pena spendere una quindicina di milioni (come minimo) per computerizzare l'edizione degli spartiti, rivolgiti a negozi più che specializzati per sapere come entrare in possesso di programmi e computer idonei allo scopo.

Primi lavori

□ **Sono un ragazzo di 12 anni che possiede un C/64 ed una miniprinter con la quale ho stampato il programma che vi invio. Se lo ritenete opportuno pubblicatelo pure.**

(Diego Trevisan - Vicenza)

• Il listato ricevuto è semplice, ben impostato e dimostra che, a dispetto della tua giovane età, stai imboccando la strada giusta per la realizzazione di programmi ben più complessi.

Putroppo lo ritengo poco adatto alla pubblicazione non solo per la sua applicazione (piuttosto particolare) ma, soprattutto perchè, qui in Redazione, non abbiamo il tempo di digitare i listati che i lettori spediscono.

I programmi che vedi pubblicati, infatti, sono stati in precedenza controllati uno per uno ed il tempo necessario per verificarne il corretto funzionamento è di certo considerevole.

A stento, quindi, riusciamo a star dietro ai programmi inviati su supporto magnetico dal caricamento immediato: figurarsi se è possibile trovare una pausa per digitare programmi che pervengono su carta!

Se, in un futuro che ti auguriamo prossimo, realizzerai qualcosa di interessante, ti prego di telefonarci PRIMA di inviare la cassetta (oppure il disco) contenente il programma che desideri veder pubblicato; in caso contrario, come putroppo è capitato ad altri lettori, rischi di lavorare su un argomento non ritenuto idoneo, oppure già in corso di pubblicazione.

Capire il Sistema Operativo

□ **Esiste un libro in italiano in cui vengono commentate tutte le routine del Sistema Operativo? In caso negativo, perchè non lo pubblicate voi, magari a puntate?**

(Marcello Artioli - Ferrara)

• Il Sistema Operativo (S/O) è quell'insieme di sottoprogrammi, scritti in Linguaggio Macchina (L/M), necessari per la corretta gestione del computer.

Per quanto mi risulti, non esiste un libro del genere con commenti dettagliati in italiano e, se non erro, nemmeno in inglese.

I volumi su questo argomento, che di norma si trovano in libreria, sono infatti scritti in inglese, a patto di considerare "scritti" un termine appropriato: vi sono, infatti, soltanto stringatissimi commenti alle varie routine e non vengono riportate, tra l'altro, nemmeno le indicazioni su come utilizzarle. Totale, tra l'altro, è l'assenza di esempi o applicazioni pratiche.

Il motivo di tale schematicità è presto detto: chi desidera studiare da vicino il S/O è una persona di indubbie

capacità che conosce in maniera molto approfondita il linguaggio macchina e, quasi certamente, la lingua inglese. Le telegrafiche informazioni sulle numerosissime routine (diverse centinaia), sono quindi più che sufficienti per chi è realmente interessato.

Chi, viceversa, non ha intenzione di studiare (perchè di STUDIARE si tratta) il S/O di un calcolatore, non verrà certo allettato dalla proposta di un libro, inevitabilmente voluminoso (e costoso), anche se scritto in italiano, che illustri le potenzialità del computer in suo possesso.

In definitiva: nessuno pubblicherà mai un libro del genere che tu richiedi, a meno di rischiare un completo insuccesso commerciale. Libri sul L/M, invece, ve ne sono parecchi, alcuni ottimi, altri un po' meno.

Di tanto in tanto anche noi pubblichiamo argomenti legati al S/O ma non pensiamo, almeno per ora, di affrontare l'argomento in maniera sistematica.

Se, comunque, ti senti attratto dal linguaggio macchina, ti consigliamo di acquistare, in edicola, il fascicolo "Commodore Speciale" della Systems Editoriale, che contiene, oltre ad un breve "corso" sul L/M, ricco di esempi ed applicazioni, anche il disassemblato, commentato, di un tool grafico per il Commodore 64.

Paddle

□ Che cosa sono le paddle?
(Alessio Carretta - Segrate)

• Le Paddle, vendute in genere a coppia, costituiscono un accessorio utile come strumento di Input. Ciò significa che, a seconda della rotazione di una manopola, è possibile "comunicare" al calcolatore l'intenzione di spostare il personaggio di un videogioco oppure la scelta di una particolare opzione e così via.

Purtroppo, a dispetto della modesta cifra richiesta (circa 15000 lire) pochi programmi sono predisposti per servirsi del comodo accessorio e la loro diffusione risulta, di conseguenza, piuttosto limitata.

Naturalmente non serve a nulla collegare le paddle e cercare di servirsene in un programma che non ne preveda l'uso: sarebbe come utilizzare un televisore a colori nel tentativo di vedere, a colori, un vecchio film trasmesso in bianco e nero!

Autoboot universale

□ Ho trovato interessante l'articolo "Autoboot 128" pubblicato su C.C.C. N.35 e mi chiedo se non sia possibile caricare con il C/128 un programma per C/64 in modo da utilizzare la procedura di Autoboot, esclusiva del C/128, anche per programmi del C/64.
(Marco Morsone - Torino)

• Per Autoboot si intende quella particolare procedura automatizzata grazie alla quale, inserendo un dischetto nel computer ancora spento, sia caricato un certo programma ivi contenuto non appena si accende il calcolatore.

Come si può intuire tale procedura è comoda per gli utenti inesperti che sono in grado di utilizzare soltanto programmi professionali a prova di errore.

Un altro motivo per cui, spesso, si ricorre all'autoboot, è dovuto al desiderio di inserire particolari routine che rendano inattive alcune funzioni e che, in generale, impediscano la copia del dischetto.

Per quanto abbiamo detto risulta chiaro che un autostart è possibile solo se la particolare circuiteria del calcolatore lo preveda; in caso contrario un autoboot a freddo è del tutto impossibile.

Abbiamo pubblicato, in passato, numerosi articoli che consentano l'autostart, ma tutti richiedono la digitazione di un comando, pur se brevissimo (come, ad esempio, LOAD"*",8,1).

Caricare automaticamente un programma del C/64 in modo 128 presupporrebbe le seguenti fasi:

- Inserire, nel dischetto, la routine di caricamento automatico (per C/128 in modo 128).
- Fare in modo che il programma, caricato automaticamente, alteri vari puntatori e modifichi gran parte della Ram (e ricopi, tra l'altro, parte della zona Rom nella Ram sottostante).
- Costringere il C/128 a passare in modo C/64 (ricorrendo, in L.M, ad un GO64 o simili).
- Abilitare le routine precedentemente predisposte ad eseguire l'autostart C/64 a caldo.

La Ram del C/128 in modo C/64 costituisce, infatti, una parte dell'intera Ram del sistema ed è possibile, quindi, scrivervi "qualcosa" anche se si è in modo 128. Quando si passa al modo 64, tale Ram non viene modificata ma viene fatta partire una routine, posta in Rom, il cui indirizzo è contenuto nelle locazioni esadecimali FFFC FFFD.

Se, prima di passare al modo 64, sposti la Rom del Basic e del Sistema Operativo in Ram e modifichi le opportune inizializzazioni, DOVREBBE (il condizionale è d'obbligo) succedere quanto tu desideri.

Il lavoro di programmazione che sarebbe necessario compiere risulta però decisamente arduo e forse non vale la pena dedicare tanto tempo per risparmiare la digitazione di un semplice comando tutte le volte che accendi il computer.

Finalità di C.C.C.

□ Chi vi scrive è un vostro fedele lettore. Siete caduti così in basso che non meritate neanche questa lettera, ma l'odio e il disprezzo che nutro per voi è troppo forte... (omissis)... La vostra rivista fa schifo per non parlare dell'Enciclopedia di Routine e s... varie dedicate a quegli scemi che, come voi... (omissis)... Ma finitela col Basic e iniziate un discorso più sofisticato: per i principianti ci sono gli arretrati (per quel che servono)... Vorrei sapere cosa ha da dire de Simone, se osa rinnegare una sola riga di questa lettera... (omissis)... Sbrigatevi a pubblicare le scuse perchè non voglio spendere altri soldi in carta igienica di pessima qualità.
(Lettera anonima)

• Questa lettera, che si commenta da sola grazie al notevole coraggio espresso dal distratto lettore che ha dimenticato di firmarla, è stata presa ad esempio non per sottrarre spazio ad argomenti più importanti ma, al contrario, per chiarire quale sia la nostra posizione.

Chi, realmente, ha passione per i computer non arriverà mai a dire che del Basic sa già tutto ed è il caso di passare a qualcosa di più sofisticato. Siamo convinti che, soprattutto nel campo dell'informatica, non si finisca mai di imparare; siamo tutti, insomma, perennemente "principianti" (in senso buono) e un qualsiasi programma, per quanto banale, si rivela idoneo ad esser sviluppato, modificato, ampliato utilizzando quel meraviglioso componente presente in ognuno di noi: il cervello.

Non tutti, lo riconosciamo, sono in grado, partendo da semplici considerazioni, di sviluppare per proprio conto elaborazioni mentali: lo dimostra ancora una volta il nostro anonimo lettore che utilizza, in modo quantomeno antieconomico, una qualità di carta per nulla idonea a svolgere operazioni di stretta intimità.

Siamo però convinti che la strada giusta sia quella di illustrare, con semplicità, argomenti a torto ritenuti complessi ed in questo siamo confortati non solo dalle numerose lettere di complimenti, ma anche da quelle contenenti garbata e civile critica da parte di lettori che ci sollecitano, anzi, a rendere ancora più semplici alcuni temi già affrontati in precedenza.

Coloro che, nonostante tutto, hanno in passato ritenuto che i contenuti della rivista incominciassero ad andar "stretti" per le proprie capacità, prima di abbandonarla del tutto hanno preferito contattarci esprimendo i loro dubbi. Molto spesso queste stesse persone sono diventate, da semplici lettori, direttamente... autori degli articoli e programmi che vedete pubblicati.

Per quanto riguarda la decisione del nostro "fedele" ed anonimo lettore, mi sento in dovere di ringraziarlo per la sua decisione di non acquistare più la nostra Rivista: il suo edicolante, infatti, generalizzando pericolosamente, potrebbe ritenere che i lettori di Commodore Computer Club siano tutti come lui.

I nostri lettori, invece, sono tutte persone educate e civili.

Per ciò che, invece, riguarda le accuse inappellabili contenute nel coraggioso manoscritto, precise questioni di stile mi impediscono, vivaddio, di rispondere a tono.

Aiuto!

Molti lettori, soprattutto coloro che sono entrati in possesso del loro primo computer in occasione dello scorso Natale, ci bombardano di lettere (al ritmo di una quindicina al giorno) chiedendo informazioni su questioni di eccessiva semplicità che sono, proprio per questo motivo, non pubblicabili.

Coloro che hanno un disperato bisogno di aiuto (la cui ansia giustifichiamo pienamente) possono telefonarci (02/84.67.34.8). Allo scopo di evitare costose telefonate interurbane, consigliamo, VIVAMENTE, di leggere con attenzione il manuale allegato al computer che, nella gran parte dei casi, contiene adeguate risposte alle domande più semplici.

Per quanto riguarda altri argomenti vi consigliamo, altrettanto vivamente, di leggere TUTTI gli articoli pubblicati sulla nostra rivista perchè molto spesso, in alcuni programmi, si ricorre a particolari tecniche di programmazione oppure si affrontano argomenti apparentemente slegati dal tema principale.

Solo se, realmente, non trovate le informazioni che desiderate, telefonateci (meglio se al pomeriggio) oppure scriveteci sottoponendo il vostro quesito.

Tenete conto, tuttavia, che in nessun caso rispondiamo privatamente, nemmeno se allegate un francobollo per la risposta o se ci assicurate (come ha proposto un lettore) di pagare l'eventuale parcella per la consulenza(!).

Non ne facciamo, infatti, una questione di denaro: la decisione di pubblicare un quesito, piuttosto che un altro, è dovuta esclusivamente a motivi di interesse generale.

Ingiustizie

□ Egregio signor direttore, le scrivo per comunicarle la mia disapprovazione sul fatto che da quando compro C.C.C (quattro mesi) non ho trovato un solo gioco per C/16: solo programmi per C/64. Non mi sembra giusto.
(Adamo Gianluca - Villafranca).

• Pubblico volentieri la tua brevissima ma (forse proprio per questo) pungente lettera nella speranza che tu, nel frattempo, abbia continuato a comprare la rivista e che, soprattutto, abbia notato il nostro sforzo nel pubblicare argomenti che riguardano da vicino lo sfortunato computer della Commodore.

Tieni conto, comunque, che molti dei listati pubblicati sono validi per qualsiasi computer e, di conseguenza, anche per il C/16.

Purtroppo non mi è possibile dedicare un elevato numero di pagine a calcolatori che hanno avuto una scarsa diffusione: tieni presente che devo pur render conto del mio operato all'editore che non vedrebbe di buon occhio, giustamente, un'inevitabile, parallela insoddisfazione degli utenti del C/64, che rimane l'home computer più venduto in Italia.

Pensa che analoghe delusioni sono sopportate pazientemente da appassionati di discipline sportive poco diffuse che vedono, loro malgrado, pagine e pagine dei quotidiani dedicate al calcio, contro qualche solitaria "colonna" che parla del loro sport preferito.

Ne approfitto per rispondere anche a coloro che non ritengono giusto l'inserimento, nella nostra rivista su nastro (Software Club) di programmi per MSX e per Sinclair Spectrum.

Ricordiamo, infatti, che anche se eliminassimo dal nastro tali programmi, lo spazio lasciato libero non verrebbe "riempito" da altri programmi per computer Commodore: non pensate che sia facile produrre software

(e soprattutto proporlo a prezzi così bassi).

Molti rotocalchi e quotidiani, che leggete abitualmente, contengono di certo rubriche che non richiamano il vostro interesse e che, per quanto vi riguarda, poterebbero benissimo essere scritte in arabo dal momento che non le leggete mai.

E' così difficile, quindi, far finta che la parte di nastro che non potete caricare contenga uno dei discorsi di Craxi, il resoconto di una giornata di Marta Marzotto oppure, più semplicemente, la pubblicità del Dash?...

Robin Hood era un pirata?

□ **Giudico interessante la vostra iniziativa "Directory" che propone molto software a basso prezzo. Però la pirateria non la si combatte solo con i prezzi, ma anche con la disponibilità dei programmi.**

Come dovrebbe comportarsi, infatti, l'utente che desiderasse venire in possesso dei nuovi programmi (presenti sul mercato americano da mesi e mesi) se non rivolgendosi ai cosiddetti "pirati"? Chi ha una copia originale di "Geos", "Newsroom", "Movie Maker" ed altri, scagli la prima pietra.

Se la stessa Commodore, o chi per essa, impostasse una seria politica di distribuzione del nuovo software originale, di informazioni varie e, naturalmente, di prezzi accessibili, forse venderebbe più hardware (e magari qualche C/128 in più). Qualcosa forse si sta muovendo (vedi il coraggioso esempio della Mastertronic, copiata ripetutamente dai pirati), ma quanto dovremo aspettare per avere gli altri programmi originali, dotati di manuali e garantiti?

Scusatemi, ma mi sono lasciato trasportare...
(Costantino Pessano - Finale Ligure)

• Ecco un discorso che non fa una grinza ma che ha un notevole difetto: è rigorosamente logico e, per quanto banale (mi scusi, signor Pessano) intelligente e validissimo sotto tutti i punti di vista. Quanto ho appena affermato, però, non ha lo scopo di esprimere un complimento per le sue facoltà mentali ma, al contrario, una doverosa manifestazione di stupore nei confronti delle Alte Menti dei dirigenti di alcune aziende che sembrano fare a gara nel darsi martellate sulle dita.

Tutte le fabbriche di computer domestici (e non) hanno capito subito che il futuro dell'azienda è nel software e si sono dati (e si danno) da fare per inserire nel proprio listino numerosi programmi adatti per tutte le esigenze. La Commodore, al contrario, presenta al pubblico un catalogo che definire scarno è un complimento.

Il motivo recondito di un simile comportamento commerciale mi sfugge totalmente; rimane, ora, un dubbio: gli odierni pirati del software sono forse gli emuli di Robin Hood che toglieva ai ricchi per dare ai poveri?

E mentre rivolgo la domanda ai lettori mi accorgo di non avere a disposizione pietre da scagliare per primo...

Listato strano

□ **Caricando alcuni programmi con il C/64 dotato di Disk drive 1541 e cartuccia Fast-disk compare, alla richiesta di LIST un messaggio simile al seguente:**
1984 SYS 2099

E' possibile eliminare il fast disk per esaminare il programma?
(Eleonora Menini - Roma)

• Finalmente una rappresentante del gentil sesso! Chissà perchè, infatti, si ritiene che i computer siano "roba" da maschi.

Tornando al quesito, ci permettiamo di ricordare ciò che abbiamo spesso scritto: se, dopo il caricamento di un programma, il comando di LIST fornisce una sola istruzione (SYS XXX o simili), quello che è stato caricato non è un programma Basic ma, al contrario, un programma in Linguaggio Macchina (oppure compilato). Ciò significa che è impossibile "estrarne" il listato per il semplice motivo che... non c'è (almeno per come comunemente si intende).

Lo "strano" messaggio che compare, quindi, non è dovuto al fast-disk o ad un errato caricamento da parte dell'utilizzatore.

Random e L.M.

□ **Come è possibile ottenere un numero casuale in ambiente linguaggio macchina?**
(Stefano Lanticina - Magenta)

• Il numero casuale che normalmente viene generato con la funzione RND si basa sull'orologio interno che, nel caso del C/64, è gestito attraverso le tre locazioni di memoria RAM 160, 161, 162 il cui contenuto viene aggiornato automaticamente ogni 60-mo di secondo. In altre parole il contenuto della locazione 162 viene incrementato da 0 a 255; il successivo 60-mo di secondo il suo valore torna nullo mentre la locazione 161 si incrementa di uno. Allo stesso modo si comporta la locazione 160 che, però, non si azzera dopo 256*256*256 sessantesimi di secondo (come saresti indotto a credere), ma dopo un periodo di tempo pari a 24 ore.

Se ti trovi in L/M, quindi, puoi fare, di tanto in tanto, un "salto" alla lettura del contenuto del byte 162 che, variabile tra 0 e 255, non dovrebbe esser prevedibile a priori a meno di effettuare una lettura ad intervalli di tempo ben precisi e sempre rigorosamente eguali tra loro. Ma anche in questo caso, effettuando sapientemente calcoli... improbabili sul contenuto delle tre locazioni, dovresti riuscire nell'intento.

Problemi con Easy script

□ **Quando uso il word processor Easy Script, e desidero utilizzare fogli singoli, ho difficoltà ad impostare i vari comandi per fare in modo che un lungo**

testo venga stampato correttamente. Potete aiutarmi?
(Tonio Gira - Galatone)

• Un qualsiasi word processor, e quindi anche Easy Script, possiede due comandi fondamentali per la gestione (detta più propriamente "formattazione") del foglio di carta: il primo riguarda la lunghezza del foglio di carta adoperato, mentre il secondo determina il numero di righe che si desidera scrivere su ciascun foglio.

Se, ad esempio, desideri inserire fogli di carta più lunghi del normale (capaci, magari, di contenere ciascuno un centinaio di righe di testo) non sei tuttavia obbligato a riempirli per intero, ma è possibile distribuire il testo su un numero maggiore di pagine di quanto sia strettamente sufficiente.

Desiderando utilizzare fogli di carta fuori formato, sarà comunque necessario effettuare alcune prove per controllare quante righe è possibile stampare senza che il sensore di fine carta interrompa la procedura.

Easy Script, non appena viene attivato, "ritiene" che i fogli utilizzati siano di formato tale da poter stampare 66 righe di testo ed inserisce automaticamente un gruppo di linee vuote tra una pagina e la successiva in modo da "saltare" la parte zigrinata che facilita lo strappo nei fogli continui.

Se, nelle prove, ti accorgi che il numero di 66 righe comporta inconvenienti per il formato di carta che adoperi, puoi modificarne il valore, portandolo, ad esempio, a 60. Il modo di procedere è quello che già conosci: dopo aver premuto il tasto funzione F3 (che fa comparire un asterisco in campo inverso) digita il codice di lunghezza pagina (Page Length) seguito dal numero di righe desiderato.

Esempio:

*pl60

Ti ricordo che è possibile migliorare l'aspetto di un documento impostando anche (Text Length) il massimo numero di righe stampabili, che deve essere, come è intuitivo, inferiore o al massimo eguale a quello precedente. Esempio:

*tl50

In ogni caso, anche prima di effettuare le prove su carta, ti consiglio di ricorrere alla simulazione su video di ciò che verrà realmente stampato: premi il tasto funzione F1 e, subito dopo, i tasti "O" (che significa Output) e "V" (che sta per Video).

In tal modo il video si trasforma in una "finestra" da far scorrere da destra a sinistra con i tasti cursore, dall'alto in basso premendo la barra spaziatrice (al contrario è impossibile). Alla fine della pagina, premendo il taso "C", si potranno esaminare quelle successive, fino all'ultima. Per tornare al modo testo è sufficiente premere il tasto Run/Stop in qualsiasi momento.

TOT 13 non funziona?

□ Ho trascritto il programma TOT 13 pubblicato su C.C.C. N.35 ma a volte, impostando alcuni sistemi, il programma fornisce elaborazioni che a mio parere non sono esatte. In particolare ottengo addirittura zero colonne impostando il seguente sistema formato da cinque doppie, una tripla e sette fisse:

1X,1X,1X,1X,1,1X,X,X,1X2,2,2,1,1

Le altre impostazioni, necessarie per far girare il programma, le ho fissate come segue:

Numero di segni:

1 da 2 a 4
X da 1 a 3
2 da 1 a 2

Consecutività segni:

1: 4
X: 2
2: 2

Perchè il programma non funziona?
(Dario Pisanello - Treviglio)

• Il programma, anzi, fornendo una risposta nulla, dimostra di funzionare perfettamente dal momento che non esistono colonne del sistema indicato che soddisfano, CONTEMPORANEAMENTE, tutte le condizioni impostate.

Per meglio seguire il ragionamento ti consiglio di digitare e far girare a parte il seguente programma che non è altro se non una modifica del listato pubblicato sullo stesso N.35 a pagina 22 (secondo programma):

```
120 REM SVILUPPO DEL SISTEMA
TOTOCALCIO
130 REM  1X,1X,1X,1X,1,1X,X,X,1X2,
2,2,1,1
140 A$(0)="X":A$(1)="1":A$(2)="2":
A$(3)="X":W=1
150 FORA=0TO1:REM 1/X
160 FORB=0TO1:REM 1/X
170 FORC=0TO1:REM 1/X
180 FORD=0TO1:REM 1/X
190 FORE=0TO1:REM 1/X
191 FORF=0TO2:REM 1/X/2
195  PRINTA$(A)A$(B)A$(C)A$(D)"1"
A$(E)"XX"A$(F)"2211";
200 PRINTCHR$(18)W:W=W+1
210 NEXTF,E,D,C,B,A
```

Questo programma visualizza le 96 colonne (=2*2*2*2*2*3, cinque doppie ed una tripla) del sistema sviluppato integralmente.

Ti accorgerai facilmente che le prime 88 colonne (ad eccezione della 47, 48, 71, 72, 83 e 84) non soddifano la condizione della quantità massima dei segni "X" (fissata da 1 a 3) e vengono, pertanto, scartate. Ma anche le colonne indicate tra parentesi vengono scartate perchè il numero di segni "1" presenti supera quello impostato (da 2 a 4). Le rimanenti colonne (dalla 89 alla 96) vengono scartate per lo stesso motivo.

E' quindi inevitabile che l'elaborazione non fornisca colonne giocabili!

Ti consigliamo, pertanto, di studiare con la massima attenzione le condizioni "ai limiti" in modo da evitare, per il futuro, delusioni o sorprese inaspettate.

Defender of the Crown

Solo per l'Amiga poteva essere scritto un adeventure grafico di notevole effetto

di Giovanni Valli & Ornella Fidani

Chiunque possegga un Commodore Amiga e, desideri usarlo nei momenti di svago, da oggi avrà un serio motivo in più per essere soddisfatto della fiducia accordata al nuovo computer della Commodore.

Il motivo si chiama "Defender of the Crown": già alcuni lo chiamano "il primo grande gioco per l'Amiga" e, senza essere così categorici, dobbiamo ammettere che questo prodotto rappresenta un concreto passo in avanti rispetto alla massa dei programmi disponibili sul mercato.

Innanzi tutto risulta molto difficile collocarlo in un ambito ben definito; il suo svolgimento ricorda vagamente quello del famoso gioco da tavolo "Risiko" e lo scorerre delle immagini fa pensare ad un cartone animato, anzi, quasi ad un film, in cui chi gioca diventa il protagonista, impersonando uno lord sassone.

Abbiamo riscontrato anche un piccolo inconveniente dovuto al fatto che, se non si desidera interrompere la continuità degli avvenimenti, è necessario possedere due disk-driver: il programma, infatti, è ripartito su due dischetti.

Passiamo ora a parlare più concretamene dell'argomento del gioco.

Siamo nell'anno 1149 nell'Inghilterra travagliata da una profonda crisi politica; al ritorno da una delle Crociate il re affida le varie contèe in parte a signori sassoni ed in parte signori normanni. Ma i due popoli, come è noto, si odiano e, se non bastasse, in seno al primo dei due vi sono anche elementi di discordia. La situazione precipita quando il re viene assassinato: scoppia una guerra in cui tutti sono nemici di tutti. Sarà compito del protagonista riunificare il regno sotto le proprie redini.

Questa introduzione, scritta su eleganti pergamene ed accompagnata da suggestive immagini, precede il vero e poprio inizio del gioco, insieme alla presentazione dei personaggi (i Sassoni, i Normanni, le dame) ed alla scelta del protagonista all'interno di una cerchia composta da quattro cavalieri; questi ultimi posseggono determinate caratteristiche per quanto riguarda il modo di comandare, la loro abilità nei tornei e nel gioco di spada.

Una volta terminata la parte preliminare, vediamo il ritratto del nostro campione sullo sfondo dell'isola britannica. L'immagine è di grande effetto coreografico perchè al mare che circonda l'isola è impresso un movimento molto realistico. Sulla sinistra dello schermo troviamo, in un trafiletto, l'indicazione dei crediti, dell'introito mensile e del mese dell'anno in cui ci troviamo.

Ogni "mossa" del giocatore corrisponde al periodo di tempo di un mese durante il quale aumenteranno i nostri crediti.

Successivamente compare la cartina geografica dell'Inghilterra suddi-

Entra nel grande Club

Fin dallo sbarco in Italia della Commodore ***Commodore Computer Club*** è il punto di riferimento di tutti gli utenti di C/64, Vic 20, C/16, Plus 4 ed ora di PC 10/20 ed Amiga.

Articoli didattici, recensioni e programmi istruttivi ed a basso costo hanno fatto di *Commodore Computer Club* la prima rivista italiana d'informatica.

Ma, per i lettori, Commodore Computer Club non è solo rivista: è consulenza telefonica gratuita, software originale pubblicato a latere dalla stessa casa editrice, un ponte verso l'informatica "maggiore" anche attraverso la collaborazione con le riviste sorelle *"Personal Computer"* e *"Computer"*.

E' per questa ragione che, anno dopo anno, aumenta il numero dei lettori che preferiscono ricevere la rivista in abbonamento invece di acquistarla in edicola. Ad essi l'editore riserva una serie di vantaggi esclusivi come:

- **un libro in omaggio** da scegliere tra i titoli disponibili della collana I libri di Systems*;
- **l'uso di una linea telefonica speciale** per richieste di consigli, e consulenza, il cui numero e le modalità d'uso verranno comunicate in forma riservata alla ricezione dell'abbonamento;
- **un canone annuo particolarmente interessante** di lire 40.000 per 11 fascicoli di Commodore Computer Club e di lire 35.000 per 11 fascicoli di Personal Computer;
- **l'esclusivo canone cumulativo** di lire 65.000 per 11 fascicoli di Commodore Computer Club ed 11 di Personal Computer;
- **uno sconto del 10%** su tutti gli acquisti per corrispondenza dei prodotti software su disco o cassetta, fascicoli arretrati o libri della Systems senza limiti di quantità.

** I titoli disponibili sono quelli reclamizzati sull'apposita pagina pubblicitaria "La libreria di Systems".*

Inviatemi in omaggio il volume della collana i libri di Systems..............................

Registrate oggi stesso il mio abbonamento a: ☐ Commodore Computer Club (Lire 40.000)
☐ Commodore Computer Club+Personal Computer (Lire 65.000)

☐ Desiderando ricevere le copie ordinate con la massima urgenza, accludo assegno bancario n.ro..............................
Banca.............................. per lire.............................. voi intestato.

☐ Contentandomi dei normali tempi postali ho inviato oggi stesso l'importo di lire .... a mezzo C/C postale N. 37952207 intestato a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Nome ....
via .... **N.ro.** .... **telefono** ....
CAP .... **Città** ....

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

visa in diciotto territori, di cui sei sono occupati dai castelli dei contendenti e dodici sono ancora da occupare. In un angolo sono ben evidenziate le scelte disponibili:

- *Partecipa ad un torneo*
- *Parti per una conquista*
- *Fai un raid (in un castello nemico)*
- *Raggruppa un esercito*
- *Leggi la mappa*

Tutte, tranne le ultime due, fanno passare un mese di tempo se selezionate; inoltre, contemporaneamente alle nostre azioni, gli avversari si arricchiscono, guadagnano un territorio al mese e combattono. Esaminiamo ora quello che compete al giocatore.

La prima possibilità è quella di partecipare ad un torneo contro altri cavalieri; al vincitore spetta la gloria oppure il territorio messo in palio dal perdente. La realizzazione grafica della sequenza è, a dir poco, spettacolare: in particolar modo la vista dell'avversario che si avvicina cavalcando il destriero, con la lancia puntata, è impressionante.

Per vincere lo scontro dobbiamo dirigere la lancia con il mouse, operazione, questa, veramente difficile; più facile è invece portare a termine un raid, trafiggendo con la spada tutti i guerrieri che si parano innanzi per bloccarci la strada del tesoro del nemico.

Per conquistare un territorio si deve, per prima cosa, formare un esercito, comprando, con i propri averi, fanti, cavalieri e catapulte. Successivamente si sposta il mouse sulla scelta "Parti per una conquista" e si decide quale parte dell'esercito rimarrà a casa e quale, invece, partirà per la campagna. Per formare un'armata più potente bisognerà attendere che quella attualmente impegnata ritorni a casa; si sceglie quindi il territorio da invadere e, se in quest'ultimo è presente un avversario, ha luogo un combattimento.

Possono capitare eventi imprevisti, come ad esempio la possibilità di intervenire nel salvataggio di una dama rapita dai Normanni, oppure l'attacco nemico al proprio castello. Il primo, se va a buon fine, dà modo di assistere ad una romantica scena d'amore, mentre il secondo porta alla fine del gioco, se non si dispone di una numerosa guarnigione di difesa.

Concludiamo il nostro discorso su "Defender of the Crown", sottolineando il fatto che la musica contribuisce ancora di più a render partecipe il giocatore dell'epoca in cui si svolge il gioco.

Il futuro dell'utente Commodore

Un'intervista con Michele Di Pisa, della Systems Editoriale, responsabile della nuova impostazione tecnica delle riviste di informatica

D/ Ultimamente il contenuto delle riviste della Systems Editoriale sembra cambiato. Qual è il motivo?

R/ Una casa editrice specializzata, qual è la nostra, non può restare insensibile alle variazioni del mercato, inteso questo sia come reale andamento delle vendite software e hardware sia come aspettative degli utenti. In questi ultimi anni il parco macchine, dal settore tipicamente professionale cui erano destinate, ha interessato anche l'utenza cosiddetta domestica con rapidità forse eccessiva. Alcuni hanno parlato di boom (inizio anni '80) e, ora, di riflusso. Ma non è esattamente una questione di moda che determina il minor volume di vendite che oggi di riscontra.

D/ A che cosa, dunque, attribuisce la stasi del mercato che sembra interessare il mercato dell'home computer?

R/ Anzitutto al fatto che l'utente è maturato con una rapidità maggiore di quanto ci si aspettasse. Giustamente ci si è accorti che un computer, benchè potente, senza software non serve a nulla; contemporaneamente ci si è accorti che imparare a programmare non è così semplice, come sembrava promettere certa pubblicità. In breve l'utente "medio" si è ritrovato in casa un calcolatore capace solo di fare videogame e lo ha quindi abbandonato. La successiva diffusione di riviste di informatica, su supporto magnetico in particolare, ha risvegliato il suo interesse, ma spesso rimane deluso a causa del tipo di software facilmente rintracciabile, vale a dire videogiochi. Ecco dunque che la Systems Editoriale propone, nelle sue riviste, articoli "alternativi" più idonei a sviluppare un discorso informatico in senso lato.

In secondo luogo, la potenza offerta dai moderni computer a 16 bit e, soprattutto, l'attuale basso prezzo d'acquisto di tali macchine, ha risvegliato l'interesse sia del "vecchio" utente, abituato alla modesta potenza di un otto bit, sia di coloro che, finora, hanno preferito rinviare l'acquisto di un computer. Entrambi gli utenti sono comunque molto prudenti e preferiscono documentarsi parecchio prima di effettuare l'acquisto dell'hardware.

D/ In che modo la Systems Editoriale pensa di risolvere i problemi degli utenti (reali o potenziali che siano)?

R/ Anzitutto non possiamo dire che risolviamo i problemi, e ciò non per falsa modestia ma perchè la casistica delle esigenze è talmente ampia che è impossibile essere esaurienti su temi specifici.

Per ciò che riguarda le varie fasce di utenza, comunque, possiamo dire che la rivista "Computer" si rivolge ai professionisti e alle piccole aziende che intendono fare il "gran passo" introducendo un elaboratore nello studio professionale o nell'azienda. Grazie ai famosi benchmark su apparecchi in prova, a volte addirittura impietosi, "Computer" si rivolge anche a coloro che già possiedono un computer professionale e desiderano aumentarne la capacità produttiva acquistando un nuova unità di memoria, una stampante, un monitor, un secondo terminale e così via.

Per quanto riguarda la fascia "bassa", (e in Italia significa quasi esclusivamente Commodore) la Rivista "Commodore Computer Club" si rivolge, ma non esclusivamente, al pubblico giovane e giovanissimo grazie all'elevato contenuto didattico di primo (Basic) e secondo livello (Linguaggio Macchina). Un continuo dialogo con i lettori rende proficuo ogni articolo della rivista.

D/ Non le pare che tra le due fasce appena menzionate vi sia un vuoto?

R/ Recenti inchieste di mercato, condotte presso utenti di computer domestici, hanno confermato il desiderio di evolversi verso applicazioni e sistemi superiori.
Un'elevatissima percentuale di utenti desidera passare a macchine più sofisticate ma, contemporaneamente, sente l'esigenza di informarsi di più prima di procedere all'acquisto. Questa stessa fascia di utenza, inoltre, desidera entrare in possesso di software "professionale" e, comunque, diverso dal solito videogame. La rivista "Personal Computer" è nata proprio con il proposito di soddisfare tali esigenze e ritiene di svolgere il compito non solo pubblicando software di un certo impegno ma, soprattutto, proponendo articoli di informazione generale atti a mettere in luce le caratteristiche più salienti delle macchine che si intendono acquistare. La rubrica Ms-Dos, ad esempio, è pubblicata proprio per questo motivo. Tra breve verranno affrontate le tematiche relative a confronti tra varie macchine a 16 bit, recensioni di software, consulenze sul corretto uso di programmi professionali e tutto quel bagaglio di informazioni utili per definire la scelta dell'hardware. Contemporaneamente la parte della rivista dedicata al software (da digitare o da richiedere su supporto magnetico) accontenta l'utenza che intende restar fedele alla macchina che già possiede.

D/ A chi è affidato il compito di sovrintendere agli argomenti Commodore?

R/ La rivista "Commodore Computer Club" è da tempo sotto la direzione di Alessandro de Simone; recentemente si è deciso di affidargli anche la sezione Commodore di "Personal Computer" sia per evitare sovrapposizioni di argomenti sulle due testate, sia per garantire il mantenimento della forma "colloquiale" alla quale è abituato il lettore di "Commodore Computer Club" che desidera evolversi verso l'alto. Possiamo affermare con giusto orgoglio che l'utente Commodore non ha bisogno di acquistare altre riviste oltre quelle edite dalla Systems Editoriale. L'ultimo handicap era infatti costituito dalla carenza di software proposto su dischetto, ma recentemente, con "Directory", anche questo scoglio è stato superato e procurarsi buon software a basso prezzo non è più un sogno.

D/ Che consigli, quindi, può dare all'utente Commodore?

R/ Systems Editoriale, naturalmente, per non aver bisogno d'altro per il suo computer, presente e futuro.

Un ritardo promettente

Se i numeri ritardatari del Lotto decidessero di uscire insieme, darebbero ragione ad una delle numerose teorie nate nel tentativo di sbancare lo Stato...

di Salvatore Di Guida

Nel gioco del Lotto sono 90 i numeri che possono essere estratti su ogni ruota; quindi esistono 90*89/2=4005 ambi diversi che si possono formare con i predetti numeri.

Da un attento esame delle ultime estrazioni rileviamo che tra i numeri estratti, su una determinata ruota, figurano almeno due numeri compresi tra 1 e 49. Nel caso in cui ciò non avvenga, il ritardo massimo, per una coppia di tali valori, è in genere di appena due estrazioni. Possiamo perciò limitarci, nella trattazione che segue, a considerare i soli numeri da 1 a 49: il numero degli ambi si riduce così a 49*48/2=1176.

Volendoli giocare al Lotto occorrono 1176 bollette, una per ogni ambo; difficilmente però troveremo un impiegato del banco Lotto disposto a compilarle!

Da qui la necessità di ricorrere ad un sistema, a "garanzia" dell'ambo, grazie al quale i numeri da 1 a 49 vengano distribuiti in 56 gruppi, ciascuno di sette numeri (onde il nome di SETTUPLA dato ad un gruppo) in modo che siano presenti tutti i 1176 ambi, nessuno dei quali è ripetuto e nessuno escluso.

Poichè è possibile giocare sette numeri su una sola bolletta (non tutti sanno che su una stessa bolletta possono essere messi in gioco fino a 20 numeri!), basterà compilarne 56, una per ogni settupla, per avere la garan-

zia dell'ambo nel caso della sortita, su una determinata ruota, di due numeri compresi tra 1 e 49.

Non volendo, ovviamente, giocare tutti i 56 gruppi, ricorriamo al programma "AMBI(1-49)" che permette la ricerca di quelle settuple che tardano a dare l'ambo su una o più ruote.

Come usare i tre programmi

Prima di iniziare qualsiasi operazione è indispensabile salvare su disco come file sequenziale le 56 settuple costituenti il sistema: a ciò è abilitato il programma "SETTUPLE PRG" e tale operazione va effettuata una sola volta.

Riconosciamo che, in effetti, sarebbe più... "informatico" determinare l'algoritmo che genera i numerosi valori (cioè le settuple) contenuti nei DATA; ne approfittiamo per girare il problema a qualche lettore volenteroso che riesca nell'intento tenendo conto di quanto è stato detto in precedenza (magari non limitandosi solo ai primi 49 numeri).

Sullo stesso disco occorre poi salvare le estrazioni che si vogliono esaminare, per esempio tutte quelle avvenute nel 1986, utilizzando il programma "ESTRAZIONI".

Di settimana in settimana, servendosi, appunto, di "Estrazioni", salveremo tutte quelle avvenute.

Dando il Run al programma "AMBI(1-49)", infine, il computer legge da disco, e memorizza, le 56 settuple (registrate grazie a SETTUPLE PRG). Quindi è possibile scegliere tra le seguenti opzioni:

U: uscita programma
E: esame estrazioni
V: settuple su video
S: settuple su stampante.

Con la seconda opzione viene chiesto quante e quali estrazioni (memorizzate in precedenza su disco) si vogliono esaminare e sottoporre ad elaborazione. E' quindi necessario stampare a parte la directory o, comunque, prenderne nota, prima di far partire il programma.

Il numero massimo di estrazioni che si possono esaminare, in una sola volta, è 100, ma variando opportunamente il dimensionamento del vettore E$, si potrebbero esaminare, sempre in una sola volta, tutte le estrazioni avvenute fino ad oggi.

Dopo aver introdotto anche le date delle estrazioni, nella forma giorno-mese-anno, viene chiesto il numero delle ruote interessate: se questo è minore di 10, occorre precisare quali, introducendo 1 per la ruota di Bari, 2 per la ruota di Cagliari e così via.

Il computer procede, quindi, alla eliminazione delle settuple che hanno già dato l'ambo su una delle ruote prescelte.

Sul video apparirà, infatti, il numero delle settuple che ancora non hanno dato l'ambo, la ruota che è stata esaminata, il totale delle estrazioni esaminate e quello ancora da esaminare.

E' chiaro che aumentando il numero delle ruote deve diminuire il numero delle estrazioni da esaminare in una sola volta, altrimenti non resteranno gruppi che tardano a dare l'ambo.

Se i tempi di elaborazione vi sembrano lunghi, compilate il programma con il Pet/Speed, l'Austrocompiler o altri compilatori: i risultati, in termini di tempo, vi risparmieranno noiose attese.

Quanto si può vincere?

Diciamo infine qualcosa sulle somme che si possono vincere: non sono alte ma, in compenso, si possono realizzare spesso.

Giocando una settupla su una determinata ruota e tenendo conto che con sette numeri si possono ottenere 21 ambi, 35 terni, 35 quaterne e 21 cinquine, si ha una vincita pari a: 11,90 volte la posta per l'ambo; 121,42 volte la posta per il terno; 2285,71 volte la posta per la quaterna ed infine 47619,04 volte la posta per la cinquina.

Coloro che hanno difficoltà a rintracciare la "storia" delle estrazioni del Lotto possono mettersi in contatto con l'autore. Ah, dimenticavamo: BUONA FORTUNA!

```
100 REM *** A M B I(1-49) ***
110 :
120 REM ***      AUTHOR       ***
130 REM ***     SALVATORE     ***
140 REM ***     DI  GUIDA     ***
150 REM ***    0823/469775    ***
160 :
170 DIM S(56,7),MU(10,5),ES$(10
    0)
180 PRINT"[CLEAR]":POKE 53280,0
    :POKE 53281,0:PRINT"[RVS][G
    IALLO]ATTENDERE PREGO![RVOF
    F]"
190 OPEN 1,8,0,"SETTUPLE,S,R"
200 FOR I=1 TO 56:FOR J=1 TO 7:
    INPUT#1,S(I,J):NEXTJ:NEXTI
210 CLOSE 1:PRINT"[CLEAR]"
```

```
220 GOTO 560
230 PRINT"[CLEAR]"
240 W=0:T=0
250 FOR I=1 TO 56:FOR X=1 TO 5:
    FOR J=1 TO 7
260 IF S(I,J)=0 THEN 340
270 IF S(I,J)=U(X) THEN K=K+1
280 NEXTJ
290 NEXTX
300 IF W=1 AND K<2 THEN GOSUB 3
    80:GOTO 340
310 IF W=2 AND K<2 THEN GOSUB 9
    00:GOTO 340
320 IF K<2 THEN T=T+1:GOSUB 470
330 IF K>=2 THEN GOSUB 450
340 K=0:NEXTI
350 GOSUB 490
360 IF RS=1 THEN 870
370 GOTO 560
380 T=T+1
390 IF T/11<>INT(T/11) THEN 420
400 GET G$:IF G$<>" " THEN 400
410 PRINT"[CLEAR]":GOTO 420
420 FOR J=1 TO 7:PRINTS(I,J);
430 NEXTJ:PRINT:PRINT
440 RETURN
450 FOR J=1 TO 7:S(I,J)=0:NEXTJ
460 RETURN
470 PRINT"[RVS]NUMERO SETTUPLE
    NON ELIMINATE[RVOFF][2 DOWN
    ]"T:PRINT"[4 UP]"
480 RETURN
490 IF W=1 THEN GET G$:IF G$<>"
     " THEN 490
500 PRINT"[CLEAR]":PRINT"[4 DOW
    N][RVS]RESTANO:[RVOFF]"T"SE
    TTUPLE":IF T=0 THEN  GOTO 9
    30
510 IF W=1 OR W=2 THEN Y=R(NR)
520 PRINT"[UP][20 RIGHT]"Y"•RUO
    TA ESAMINATA"
530 PRINT:PRINT"TOT.ESTR."TE;
540 PRINT"[4 RIGHT]ESTR. DA ESA
    MINARE"EE:PRINT
550 RETURN
560 PRINT:PRINT"[2 DOWN][RVS]U)
     USCITA PROGRAMMA[RVOFF]":P
    RINT"[2 DOWN][RVS]E) ESAME
    ESTRAZIONI[RVOFF]"
570 PRINT"[3 DOWN][RVS]V) SETTU
    PLE SU VIDEO[RVOFF]"
580 PRINT"[2 DOWN][RVS]S) SETTU
    PLE SU STAMPANTE"
590 GET H$:IF H$<>"U" AND H$<>"
    E" AND H$<>"V" AND H$<>"S"
    THEN 590
600 IF H$="U" THEN PRINT"[CLEAR
    ]":PRINT"[3 DOWN][RVS]BUONA
     FORTUNA![RVOFF]":END
610 IF H$="E" THEN GOTO 640
620 IF H$="V" THEN W=1:T=0:RS=0
    :PRINT"[CLEAR]":GOTO 250
630 IF H$="S" THEN W=2:T=0:RS=0
    :PRINT"[CLEAR]":GOTO 250
640 PRINT"[CLEAR]":PRINT"QUANTE
     ESTRAZIONI?":PRINT:PRINT"
    "NE:PRINT"[2 UP]":INPUT NE:
    EE=NE:PRINT"[CLEAR]"
650 FOR L=1 TO NE:PRINTL"•ESTRA
    ZIONE(G-M-A)":PRINT:INPUT E
    S$(L):PRINT"[CLEAR]":NEXTL
660 PRINT"[CLEAR]":PRINT"QUANTE
     RUOTE?":PRINT:PRINT"  "NR:P
    RINT"[2 UP]":INPUT NR:PRINT
670 IF NR=10 THEN FOR H=1 TO NR
    :R(H)=H:NEXTH:PRINT"[CLEAR]
    ":GOTO 700
680 PRINT"QUALI?":IF NR=1 THEN
    PRINT"[UP][4 RIGHT]E?"
690 PRINT:FOR H=1 TO NR:PRINT"
    "R(H):PRINT"[2 UP]":INPUT R
    (H):NEXTH:PRINT"[CLEAR]"
700 FOR L=1 TO NE
710 EE=EE-1:TE=TE+1
720 OPEN 1,8,0,"0:"+ES$(L)+",S,
    R"
730 FOR Y=1 TO 10:FOR X=1 TO 5
740 INPUT#1,MU(Y,X)
750 NEXTX:NEXTY
760 CLOSE 1
770 FOR Y=1 TO 10
780 FOR H=1 TO NR
790 IF Y<>R(H) THEN GOTO 810
800 GOTO 830
810 NEXTH
820 GOTO 870
830 FOR X=1 TO 5
840 U(X)=MU(Y,X)
850 NEXTX
860 RS=1:GOTO 240
```

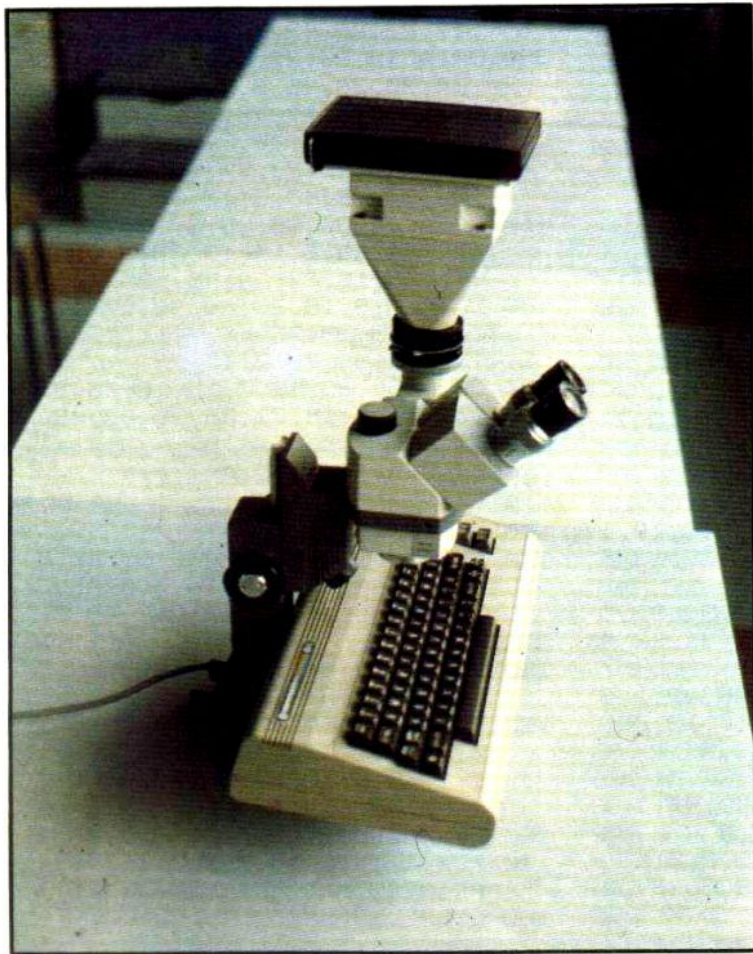
```
870 NEXTY
880 NEXTL
890 GOTO 560
900 T=T+1:OPEN 1,4:FOR J=1 TO 7
    :PRINT#1,S(I,J);:NEXTJ:PRIN
    T#1," "CHR$(13)
910 CLOSE 1
920 RETURN
930 PRINT:PRINT
940 PRINT:PRINT"OGNI SETTUPLA H
    A DATO":PRINT:PRINT"ALMENO
    UN AMBO!":END
```

```
100 REM ***  SETTUPLE PRG  ***
110 :
120 :
130 PRINT"[CLEAR]":POKE 53280,0
    :POKE 53281,0
140 PRINT"[RVS][GIALLO]INSERIRE
     DISCO NEL DRIVE![RVOFF]"
150 DIM S(57,7)
160 I=1
170 FOR X=1 TO 392
180 J=J+1
190 READ A
200 IF J=7 THEN S(I,J)=A:I=I+1:
    J=0
210 S(I,J)=A
220 NEXTX
230 OPEN 1,8,1,"SETTUPLE,S,W"
240 FOR I=1 TO 56:FOR J=1 TO 7
250 PRINT#1,S(I,J)
260 NEXTJ:NEXTI
270 CLOSE 1
280 DATA 1,2,3,4,5,6,7,8,9,10,1
    1,12,13,14,15,16,17,18,19,2
    0,21,22,23,24,25,26
290 DATA 27,28,29,30,31,32,33,3
    4,35,36,37,38,39,40,41,42,4
    3,44,45,46,47,48,49
300 DATA 1,8,15,22,29,36,43,1,9
    ,17,25,33,41,49,1,10,19,28,
    30,39,48,1,11,21,24,34
310 DATA 37,47,1,12,16,27,31,42
    ,46,1,13,18,23,35,40,45,1,1
    4,20,26,32,38,44,2,8
320 DATA 21,27,33,39,45,2,9,16,
    23,30,37,44,2,10,18,26,34,4
    2.43,2,11,20,22,31,40
330 DATA 49,2,12,15,25,35,38,48
    ,2,13,17,28,32,36,47,2,14,1
    9,24,29,41,46,3,8,20
340 DATA 25,30,42,47,3,9,15,28,
    34,40,46,3,10,17,24,31,38,4
    5,3,11,19,27,35,36,44
350 DATA 3,12,21,23,32,41,43,3,
    13,16,26,29,39,49,3,14,18,2
    2,33,37,48,4,8,19,23
360 DATA 34,38,49,4,9,21,26,31,
    36,48,4,10,16,22,35,41,47,4
    ,11,18,25,32,39,46,4
370 DATA 12,20,28,29,37,45,4,13
    ,15,24,33,42,44,4,14,17,27,
    30,40,43,5,8,18,28,31
380 DATA 41,44,5,9,20,24,35,39,
    43,5,10,15,27,32,37,49,5,11
    ,17,23,29,42,48,5,12
390 DATA 19,26,33,40,47,5,13,21
    ,22,30,38,46,5,14,16,25,34,
    36,45,6,8,17,26,35,37
400 DATA 46,6,9,19,22,32,42,45,
    6,10,21,25,29,40,44,6,11,16
    ,28,33,38,43,6,12,18
410 DATA 24,30,36,49,6,13,20,27
    ,34,41,48,6,14,15,23,31,39,
    47,7,8,16,24,32,40,48
420 DATA 7,9,18,27,29,38,47,7,1
    0,20,23,33,36,46,7,11,15.26
    ,30,41,45,7,12,17,22
430 DATA 34,39,44,7,13,19,25,31
    ,37,43,7,14,21,28,35,42,49
```

```
100 REM *** ESTRAZIONI ***
110 :
120 DIM MU(10,5)
130 PRINT"[CLEAR]":POKE 53280,0
    :POKE 53281,0:PRINT"[GIALLO
    ]DATA DELL'ESTRAZIONE (G-M-
    A)"
140 PRINT:PRINT"DA SALVARE SU D
    ISCO":PRINT:INPUT E$
150 PRINT"[CLEAR]":FOR I=1 TO 1
    0:PRINTI"■RUOTA":PRINT:FOR
    J=1 TO 5:INPUT MU(I,J):NEXT
    J:PRINT:NEXTI
160 OPEN 1,8,1,"0:"+E$+",S,W"
170 FOR I=1 TO 10:FOR J=1 TO 5:
    PRINT#1,MU(I,J):NEXTJ:NEXTI
180 CLOSE 1
```

Peek, Poke & Sys

Una nuova iniziativa per conoscere più a fondo il fedele amico elettronico

a cura di Michele Maggi

Come tutti i lettori avranno certamente notato, da qualche numero non compare più su C.C.C. la rubrica "Una riga".

Ciò è dovuto ai risultati della ormai famosa inchiesta svolta tempo fa presso i nostri lettori.

Abbiamo però deciso di dedicare quelle pagine ad argomenti richiesti da più parti.

Inizia così, da questo numero, una nuova rubrica (non necessariamente mensile) che tratterà non più di minilistati da una sola riga, ma di "Peek Poke & Sys", vale a dire dello studio di particolari locazioni di memoria, di puntatori, di vettori e di indirizzi di routine del Sistema Operativo.

Ciascuna delle locazioni che verranno esaminate sarà analizzata in dettaglio e ne verrà spiegata la funzione, eventualmente con l'ausilio di qualche programmino dimostrativo.

Un po' di terminologia

Considerando che la maggior parte dei lettori interessati a questa rubrica sarà costituita da principianti, è opportuno, prima di procedere all'analisi delle locazioni di memoria, chiarire che cosa si intende per vettori, Peek, Poke, Sys e Sistema Operativo (S.O.).

Ogni locazione di memoria (byte) può contenere un valore compreso tra 0 e 255 e in funzione di ciò che

ALLARME ROSSO (per i principianti)

Nel digitare righe di programma basic che contengono istruzioni DATA, è piuttosto facile incorrere in errori di digitazione. Supponiamo che un'ipotetica linea basic numerata con 1200 debba contenere i tre valori: 123, 456, 789. Ecco alcuni esempi di errori più frequentemente commessi:

1200 DATA,123,456,789

C'è una virgola dopo la parola "DATA". I dati letti dal computer sono, in questo caso, quattro: 0, 123, 456, 789. Se, infatti, non figura alcun carattere dopo l'istruzione DATA, automaticamente viene assunto il valore nullo (0).

1200 DATA 123,456,789,

In questo caso, dopo il numero 789, il computer, grazie alla presenza della virgola erroneamente inserita, "crede" che ci sia un altro valore e, non trovandolo, lo assume come nullo (0).

1200 DATA 1234,56,789

La virgola è posizionata male, vale a dire dopo il carattere "4" e non dopo il carattere "3". Il computer non può sapere se il valore esatto è 123 oppure 1234 e individuare un errore, in questo caso, risulta piuttosto laborioso.

ontiene, e della posizione che occupa nella memoria del calcolatore, può avere, o meno un ruolo importante.

Il C/64, come tutti i computer, ha due tipi di locazione: Rom e Ram.

I byte che formano la memoria Rom non possono essere modificati in quanto sono riservati al S.O. (vedi oltre) e ad altre routine; le locazioni Ram, invece, sono in parte necessarie al corretto funzionamento del computer, e in parte sono a disposizione dell'utente per la scrittura di programmi: possono quindi essere modificate.

E' intuitivo che se vengono modificate locazioni Ram necessarie al computer per il suo corretto funzionamento, possono verificarsi anomalie che spesso portano ad un "crash" del sistema: in questi casi, pur non provocando alcun danno materiale al calcolatore, costringono a spegnere e riaccendere l'apparecchio, perdendo tutto ciò che fino a quel momento era stato battuto.

D'altro canto è proprio la modifica di certe locazioni di memoria che consente di creare interessanti effetti; il principale scopo di questa rubrica è, appunto, quello di modificare con criterio alcune locazioni in modo da ottenere risultati utili e interessanti.

Con il termine "Vettore" (ma solo in linguaggio macchina) si intende una coppia di byte (generalmete in Ram) che indicano la locazione di memoria da cui parte una certa routine del S.O. che è "puntata" dal contenuto del primo byte più il valore del secondo byte della stessa coppia moltiplicato per 256.

Nel C/64, ad esempio, la routine di List, allocata da 42778 (in notazione esadecimale: $A71A) in poi, è "puntata" dal vettore costituito dalle locazioni di memoria 774 e 775 che contengono, rispettivamente, i valori 26 e 167.

Per capire quale indirizzo stia puntano un vettore, dobbiamo prelevare il valore del primo byte (26) e sommarlo al valore del secondo (167) moltiplicato per 256; quindi:

26+167*256 = 42778

ottenibile con la seguente istruzione:

PRINT PEEK (774)+PEEK (775) *256

Peek e Poke sono, appunto, le istruzioni Basic che permettono di interagire direttamente con le celle di memoria del C/64.

L'istruzione Peek serve per leggere il contenuto di una qualsiasi locazione di memoria, mentre Poke, al contrario, serve per scrivervi un valore (purchè si tratti di una locazione di ambito Ram).

La sintassi di Peek è molto semplice ed è la seguente:

PRINT PEEK (LOC)

dove LOC è la cella di memoria (valore compreso tra 0 e 65535, pena "Illegal quantity error") di cui vogliamo conoscere il contenuto. La sintassi di Poke è la seguente:

POKE LOC,VAL

in cui LOC è la locazione in cui vogliamo scrivere il valore VAL (compreso tra 0 e 255, pena "Illegal quantity error").

Ci teniamo a sottolineare che non è possibile modificare le locazioni che appartengono alle Rom; volendo farlo, infatti, e rileggendo, mediante Peek, la stessa locazione, ritroveremmo il valore originario e non quello scritto da noi.

La sintassi dell'istruzione SYS è ancora più semplice:

SYS (IND)

ove IND è l'indirizzo della cella di memoria di partenza di un programma in Linguaggio Macchina (L.M.) che svolge una certa funzione.

Il S.O. non è altro se non una "raccolta " di programmi in L.M. che, oltre a svolgere funzioni necessarie al C/64, possono, con alcuni accorgimenti, essere chiamate tramite SYS per particolari scopi.

Pokando qua e là

Passiamo ora a qualche esempio pratico di facile comprensione.

19: Input senza punto di domanda

Una locazione che consente un effetto interessante è la n.19 che funge da flag (deviatore elettronico) per l'istruzione INPUT.

Come tutti sanno, quando il computer incontra un'istruzione INPUT, arresta il programma finchè l'utente non risponde alla richiesta.

L'Input è caratterizzato dalla presenza di un punto interrogativo, quando ci si aspetta un dato dalla tastiera, che invece scompare quando l'Input avviene da disco o nastro.

Per ottenere un Input privo del punto interrogativo è sufficiente "pokare" il valore 64 nella locazione 19, prima di far eseguire l'Input, per poi "ripokare" il valore 0 (standard) appena eseguito l'Input; esempio:

```
10 POKE 19,64
20 INPUT "SCRIVI QUALCOSA"
;A$
30 POKE 19,0
```

186: Periferica e Filename

La locazione 186 contiene il valore dell' ultima periferica in dialogo; tramite la seguente istruzione:

PRINT PEEK (186)

sarà possibile, in funzione del numero riportato, verificare quale sia stata l'ultima periferica usata. Il valore sarà certamente uno dei seguenti: 8 (oppure 9) per il drive, 4 (oppure 5) per la stampante, 6 per il plotter e così via.

Con l'istruzione SYS 62913, poi, sarà possibile visualizzare l'ultimo nome con cui si è caricato o salvato un file.

211/214: Indirizzamento cursore

Le locazioni 211 e 214 contengono l'attuale posizione del cursore video in termini di colonna e riga.
Per indirizzare il cursore in un punto particolare dello schermo sarà opportuno pokare in 211 il valore della colonna e in 214 il valore della riga in cui vogliamo indirizzare il cursore; quindi comunicarlo al S.O. con SYS 58640.

Esempio:

```
10 POKE 211,10:POKE 214,9
20 SYS 58640
30 PRINT "COLONNA 10 E RIGA
9"
```

650: Autorepeat dei tasti

Con questa arcinota poke si ottiene l'autorepeat su tutti i tasti:

POKE 650,128

199: Reverse

Ecco una Poke che permette di visualizzare un messaggio in reverse:

POKE 199,1:PRINT "REVERSE ON"

653: Flag per tasti Shift, Ctrl, C=

Questa locazione viene modificata automaticamente dal S.O. in funzione della pressione dei tasti SHIFT, C= e CTRL

10 PRINT PEEK(653):GOTO 10

Provate a premere questi tasti, anche insieme, per vedere come cambiano i valori.

Stoplist e protezioni

Molte volte, mentre si esamina un listato, sarebbe utile poter fermare lo scroll per poi riprenderlo dallo stesso punto senza dover ricorrere al tasto STOP.

Questa miniroutine, alterando il vettore di List (774-775) consente di sospendere lo scroll di un listato finchè il tasto C= è premuto.

```
10 FOR I=0 TO 11
20 READ A:POKE 8300+I,A
30 NEXT:POKE 775,32
40 DATA 72,173,141,2,201,2
50 DATA 240,249,104,76,26,167
```

Modificando ulteriormente il vettore di List è possibile inibire o modificare l'indirizzo della routine List in modo tale che, chiedendo il listato, l'effetto che ne risulta sia diverso dal solito (e spesso letale...).

POKE 774,226:POKE 775,252 - Reset
POKE 774,68:POKE 775,166 - New
POKE 774,7:POKE 775,168 - Syntax Error
POKE 774,0 - List dei soli numeri
POKE 22,35 - List senza numeri
POKE 818,32 - Disabilita Save & Load

646: Colore cursore

Modificando la locazione 646 è possibile cambiare colore al cursore senza ricorrere ai caratteri speciali.

```
10 FOR I=0 TO 15
15 POKE 646,I
20 PRINT "COLORE CURSORE"
;I
30 NEXT
```

Restore e protezioni

Il vettore 792-793 punta all'indirizzo da cui parte la routine che viene chiamata alla pressione del tasto RESTORE; modificandone i valori si possono ottenere effetti utili per le protezioni.

POKE 792,226:POKE 793,252 - Reset

POKE 792,7:POKE 793,168 - Syntax Error

POKE 808,225 - Disabilita RUN/-STOP

Abbiamo visto insieme una ventina di esempi che, anche se non hanno una utilità intrinseca, possono rivelarsi molto comodi se inseriti in programmi complessi.

I lettori sono, come sempre, invitati a partecipare inviando su carta o supporto magnetico le loro "scoperte" che, in caso di pubblicazione, verranno compensate con prodotti Systems (cassette, libri, arretrati...).

Le collaborazioni vanno indirizzate a:

Rubrica Peek & Poke
Commodore Computer Club
Systems Editoriale

V.le Famagosta, 75
20142 MILANO

Il giardino elettronico

Non angustiatevi se vivete in una città povera di verde: ora potrete crearvi un giardino personale con il C/64

di Danilo Toma

Tranquillizzatevi, non si tratta di un corso di giardinaggio, ma semplicemente di un breve programma che consente di disegnare alberi e fiori ricorrendo ad una procedura ricorsiva.

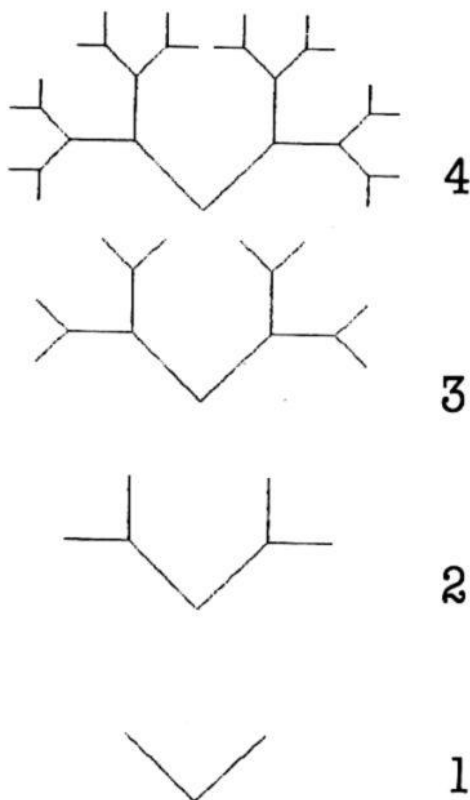
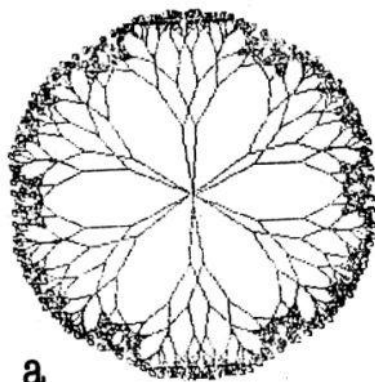
Ricordate quel programma, apparso sul n.27 della rivista, che permetteva di creare splendide figure utilizzando le curve di Peano?

Ebbene, dopo averlo scritto mi chiesi se non sarebbe stato possibile realizzarne uno che disegnasse curve altrettanto belle utilizzando strutture ad albero.

Dopo vari rinvii, dovuti a scarsità di tempo da dedicare al progetto, ecco finalmente pronto questo "sfornavegetali" elettronico. Le "varietà botaniche" che si possono ottenere (ne potete osservare alcuni esemplari riportati su queste pagine) sono veramente tante ed interessanti. L'unico cruccio che mi rimane è di non essere riuscito a trovare un algoritmo che simuli il profumo dei fiori!

Alberi

Chiarisco subito con un semplice esempio come nasce un albero del tipo che ci interessa: immaginiamo di disegnare un punto (che possiamo chiamare nodo) su un foglio. Da questo nodo facciamo partire due segmenti (o rami) divergenti e consideriamo le estremità libere dei rami stessi come due nuovi nodi da cui

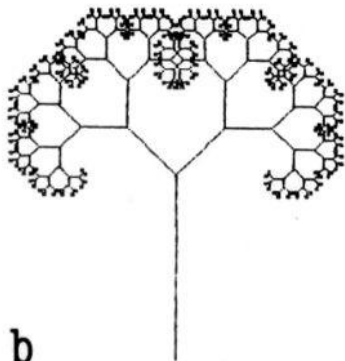
Figura 1

a

partono quattro nuovi rami (due per nodo). Dalle quattro estremità risultanti "partono" otto nuovi rami e così via. Tale processo di crescita è visualizzato, nei primi quattro stadi, nella figura 1.

Per trasformare in un algoritmo eseguibile da computer il processo descritto, ho preso carta e penna ed ho ricostruito i primi tre stadi di crescita, indicando con S il generico ramo sinistro e con D quello destro:

- **primo stadio: seq.1=S-D (cioè disegnail ramo sinistro e poi quello destro).**
- **secondo stadio: seq.2=S-(S-D)-D-(S-D) e quindi seq.2=S-seq.1-D-seq.1.**
- **terzo stadio: seq.3=S-(S-seq.1-D-seq.1)-D-(S-seq.1-D-seq.1) cioè seq.3=S-seq.2-D-Seq.2.**

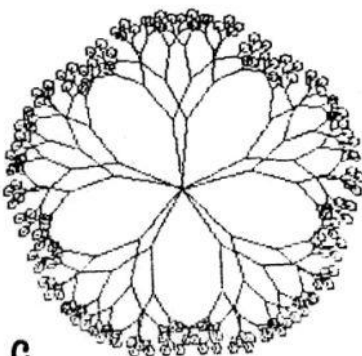
In generale quindi: seq.N=S-seq.N-1-D-seq.N-1.

Come si constata subito la sequenza base di operazioni è semplice:

b

disegna il ramo sinistro - disegna il ramo destro

però per potere tracciare il ramo destro di una qualsiasi sequenza N, occorre prima avere completato tutte le sequenze subordinate, cioè bisogna eseguire la seq.N-1 che, a sua volta, richiede l'esecuzione della seq.N-2, e così via fino alla seq.1 che, non avendo sequenze subordinate, viene finalmente realizzata interamente.

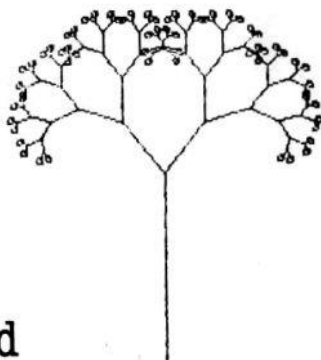
A questo punto si può completare la seq.2, cioè disegnare il suo ramo destro, e su questo costruire un'altra seq.1. Fatto ciò si passa al ramo destro della seq.3 e si riparte con il ramo sinistro di una nuova seq.2...

Mi fermo qui perchè non voglio avere sulla coscienza il suicidio di qualche lettore: chi ha i nervi saldi può benissimo continuare da solo nel tortuoso cammino e comprendere così l'algoritmo proposto nel listato (linee 580-740).

c

Il programma

Prima di tutto devo sottolineare che il programma necessita delle routine grafiche pubblicate sul n.14 per potere funzionare (oppure di quelle riportate sul fascicolo Commodore Speciale, appositamente pubblicato).

Dopo averle attivate, caricate questo listato, date il RUN e rispondete alle tre richieste del computer: nodi, angolo d'apertura dei rami, tronchi.

Il primo valore da digitare si riferisce al numero delle serie di nodi (o

d

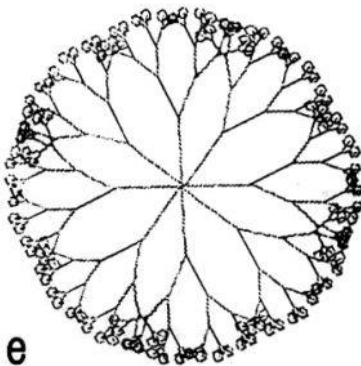
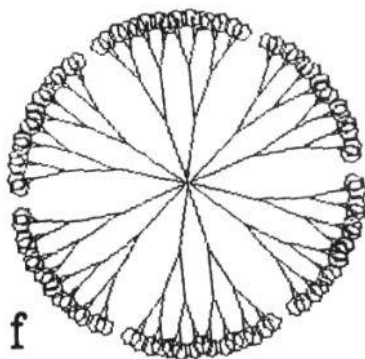
stadi o sequenze) desiderate. Ad esempio rispondendo 3 saranno disegnati i primi tre "stadi di sviluppo" dell'albero (vedi disegno 3 della figura 1).

L'angolo richiesto è quello formato dalle coppie di rami generate dai nodi e va espresso in gradi.

Infine occorre specificare se si desidera un albero "normale" (rispondendo 1) o più alberi accostati (rispondendo 2, 3, 4,...) che consentono di ottenere un fiore.

I tre valori richiesti vanno separati dalla virgola.

Per rendere più "realistici" alberi e fiori ho inserito la linea 720 che disegna una foglia esagonale alle estremità libere dei rami. In queste pagine sono riportati alcuni esempi di ciò che potete ottenere con tale programma e insieme a questi altre figure che si possono realizzare con semplici modifiche che ora illustro e che non ho inserito nel listato originale per non farvi perdere la testa con un nu-

e

f

mero eccessivo di parametri da assegnare.

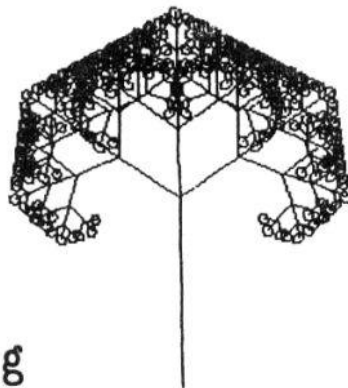
Nuovi innesti

Linea 250. La variabile P indica quanti rami devono essere generati da ogni nodo. Basta modificare tale linea così:

250 INPUT"RAMI";P

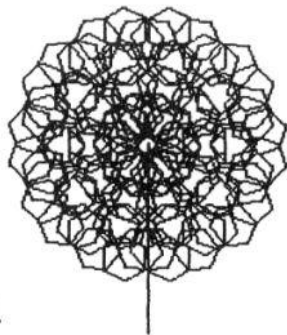
per avere la possibilità di decidere ogni volta la "prolificità" dell'albero.Ricordate che P non deve comunque essere minore di 2.

Linea 340. La variabile N1 indica quali rami vanno disegnati. Ad esempio se N1=N vengono disegnati tutti i rami, se N1=0 nessun ramo viene tracciato.

Linea 350. Per N2 (che riguarda le foglie) la spiegazione è rovesciata, ad esempio se N2=N non vengono disegnate le foglie, se N2=0 vengono disegnate foglie alle estremità di tutti i rami.

g

Anche per tali variabili si può modificare il programma in modo da decidere ogni volta:

340 INPUT"RAMI DA DISEGNARE";N1
350 INPUT"FOGLIE DA DISEGNARE";FO:N2=N-FO

Altri fattori su cui potete intervenire sono: la variabile S (linea 360) che determina di quanto si devono accorciare i rami ad ogni "stadio di sviluppo"; la variabile L (linea 370) che indica la lunghezza dei rami al primo stadio; le variabili X1 e Y1 (sempre sulla linea 370) che posizionano l'albero da disegnare.

h

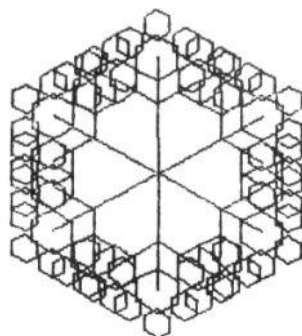
La linea 390 serve solo a disegnare il "tronco" dell'albero nel caso che abbiate assegnato valore unitario al terzo parametro richiesto dal programma.

Se volete disegnare foglie di forma e dimensioni diverse, basta modificare i parametri dell'istruzione ARC nella linea 720.

N.B. I due punti (:) presenti tra l'istruzione THEN e i comandi della grafica (DRAW, ARC, eccetera) sono indispensabili per non incorrere in un SYNTAX ERROR.

Di alcune figure riportiamo i parametri necessari per il loro tracciamento. Quali, invece, sono necessari per il disegno delle figure "D" e "L"? Tra le risposte dei lettori selezioneremo le più simpatiche e, forse, invieremo qualche libro in omaggio (a patto che abbiano indicato il titolo richiesto).

Capito, Marcia? (domanda in codice, n.d.r.).

i

SCHEDA TECNICA

Software applicativo per:

grafica
didattica

Idoneo per computer:
C/64

Adattabile ad altri computer

Richiede routine grafiche di Toma

Consigliato l'uso della stampante

Consigliato a tutti i lettori.

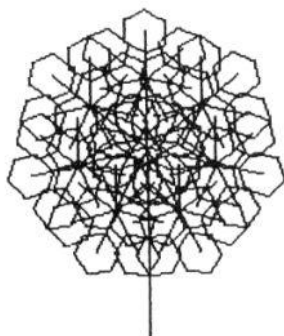
l

In questa tabella sono riportati i valori delle variabili P, N1, N2 (oltre ai valori dei parametri "base") che consentono di ottenere le figure indicate.

FIG.	NODI	ANG.	TRONCHI	P	N1	N2
G	5	110	1	3	5	4
H	2	324	1	10	0	1
I	2	308	1	7	2	1
L	X	X	X	3	2	1
M	5	50	4	2	0	0
N	4	45	5	2	0	0
O	4	50	3	2	4	0

Qui sotto sono riportate le terne di valori che consentono di ottenere i disegni realizzati con il programma base.

FIG.	NODI	ANG.	TRONCHI
A	6	50	6
B	8	90	1
C	5	60	5
D	Y	Y	Y
E	5	50	4
F	4	25	6

```
100 REM  C/64 + ROUTINE GRAFICH
    E PUBBLICATE SUL N.14 DI C.
    C.C.
160 REM  BY DANILO TOMA

190 :
200 PRINTCHR$(147)
210 POKE 53280,11
220 PRINT"NODI, ANG. D'APERT. D
    EI RAMI, TRONCHI"
230 INPUT N,GR,W
240 ←CLEAR:←GRAF6,1:←COL OR 1
250 P=2:REM *** P=RAMI GENERATI
     DA OGNI NODO
260 N=N-1:DIM  O(N),D(P)
270 REM  *** D( )=DIREZIONI DEI
     RAMI
280 D(0)=π*GR/360
290 FOR I=1 TO P-1
300 D(I)=-π*GR/180/(P-1)
310 NEXT
320 D(P)=D(0)-π
330 :
340 N1=N:REM  *** N1= NUMERO DE
    I RAMI DA DISEGNARE
350 N2=N-1:REM  *** N2= NUMERO
    DI FOGLIE DA NON DISEGNARE.
360 S=1.5:REM  *** S=FATTORE DI
     RIDUZIONE
370 X1=0:Y1=0:L=37
380 :
390 IF W=1 THEN :←DRAWX1,Y1,0,X
    1,-100,0
400 FOR J=0 TO W-1
410 AN=π/2+J*2*π/W
420 GOSUB 640
430 NEXT
440 :
450 REM  PREMI UN TASTO
460 WAIT 198,1:GET A$
470 :
480 ←TEXT6,14:PRINT
490 PRINT"ANCORA ? (S/N)"
500 WAIT 198,1:GET A$
510 IF A$="S" THEN  RUN
520 IF A$<>"N" THEN 500
530 END
540 :
550 REM ELABORA E DISEGNA
560 REM I RAMI
570 :
580 O(K)=0:IF K=0 THEN RETURN
590 I=L/S↑(K-1)
600 X1=X1+COS(AN)*I
610 Y1=Y1+SIN(AN)*I
620 AN=AN+π:K=K-1
630 :
640 AN=AN+D(O(K))
650 IF O(K)=P THEN 580
660 O(K)=O(K)+1
670 :
680 I=L/S↑K
690 X2=X1+COS(AN)*I
700 Y2=Y1+SIN(AN)*I
710 IF K=<N1 THEN :←DRAWX1,Y1,0
    ,X2,Y2,0
720 IF K>N2 THEN :←ARCX2,Y2,0,.
    5*I,.5*I,AN,AN+2*π,π/3
730 IF K<N THEN K=K+1:X1=X2:Y1=
    Y2
740 GOTO 640
750 END
```

Caleidoscopio

Un programma brevissimo dagli effetti spettacolari

di Roberto Morassi

Questo breve ma affascinante programma consente la visualizzazione di due tipi di caleidoscopi.

Il primo genera figure ricorrendo ad un gruppo di caratteri semigrafici (monocromatici) in continua trasformazione.

Il secondo disegna sullo schermo mosaici policromi, con doppia simmetria speculare.

Per passare dal primo al secondo è sufficiente premere il tasto Return in qualsiasi momento.

Poichè sul primo non si può intervenire, ci soffermeremo sulla gestione del secondo tipo di caleidoscopio.

Per non appesantire il disegno, questo viene periodicamente "traforato", sempre in modo simmetrico. Si può controllarne l'evoluzione con i seguenti tasti:

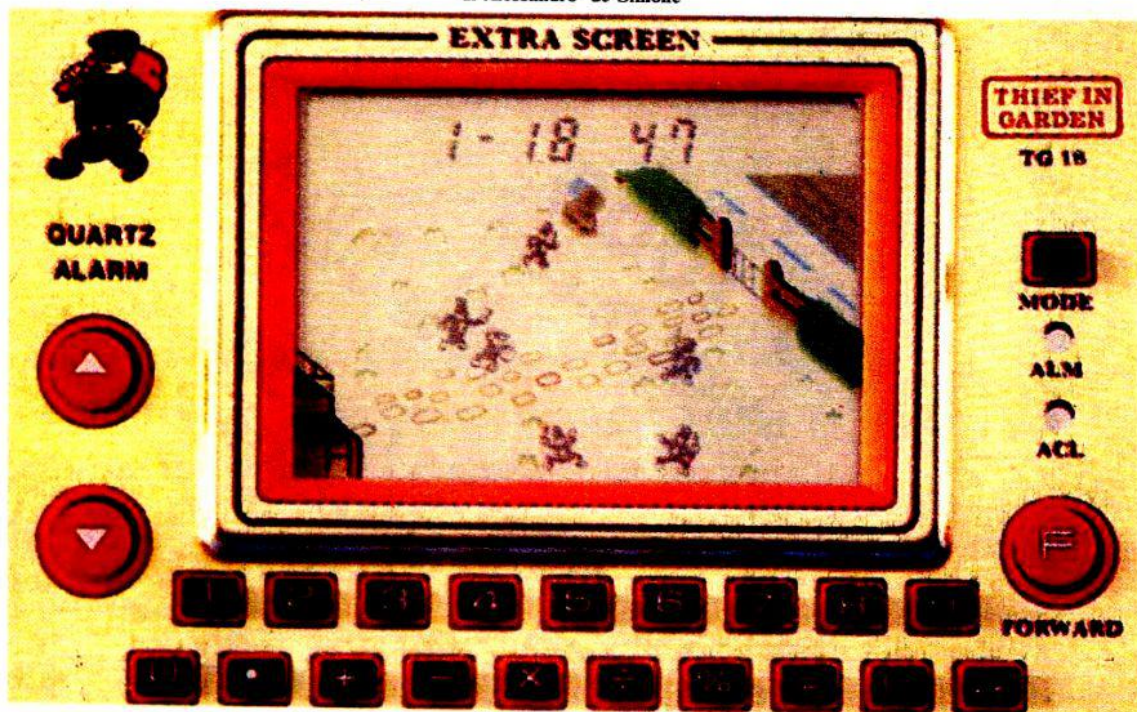
Shift: ferma temporaneamente il programma, "congelando" la schermata.
Commodore: la schermata continua utilizzando solo l'ultimo colore.
Control: la schermata continua "traforando" il disegno.

Questi ultimi due funzionano anche con lo Shift Lock inserito.

```
100 REM CALEIDOSCOPIO
120 REM BY ROBERTO MORASSI
130 REM & A.DE SIMONE
150 REM VERSIONE C/64
170 PRINT"[CLEAR]1- PRIMA VERSI
    ONE"
180 PRINT"[DOWN]2- SECONDA VERS
    IONE"
190 GET A$:IF A$="1" THEN PRINT
    "[CLEAR]": RUN220
200 IF A$="2" THEN  RUN380
210 GOTO 190
220 FOR I=55296 TO 56295:POKE I
    ,1:NEXT
230 FOR I=0 TO 7:READ CL(I):NEX
    T
240 N1=1024:N2=40:N3=.625:N4=39
    .9999
250 FOR W=3 TO 50:FOR I=1 TO 19
    :FOR J=0 TO 19
260 K=I+J:C=CL((J*3/(I+3)+I*W/1
    2) AND 7)
270 Y1=N1+N2*INT(N3*I):Y2=N1+N2
    *INT(N3*K)
280 Y3=N1+N2*INT(N3*(N4-I))
290 Y4=N1+N2*INT(N3*(N4-K))
300 POKE I+Y2,C:POKE K+Y1,C:POK
    E N2-I+Y4,C
310 POKE N2-K+Y3,C:POKE K+Y3,C:
    POKE N2-I+Y2,C
320 POKE I+Y4,C:POKE N2-K+Y1,C
330 GET Y$:IF Y$<>"" THEN  RUN3
    80
340 NEXTJ,I,W:GOTO 250
350 DATA 160,127,102,64,91,93,5
    8,32
370 REM SECONDA VERSIONE
380 X$="[RVS][NERO]
     ":REM 40 SPAZI
390 A=55304:B=55327:C=56224:D=5
    6247
400 POKE 53280,0:POKE 53281,0:P
    RINT"[NERO]"
410 Z=16*RND(0)
420 FOR X=1 TO 24:PRINTX$;:NEXT
    :FOR X=1984 TO 2023:POKE X,
    160:NEXT:POKE 198,0
430 WAIT 653,255,1:X=INT(12*RND
    (1)):Y=INT(12*RND(1)):X1=40
    *X:Y1=40*Y:W=PEEK(653)
440 T=T+1:IF T=40 THEN T=0:F=(F
    +1) AND 1:Z=16*RND(1)
450 IF F THEN Z=0
460 IF W>1 THEN Z=Z1:IF W>3 THE
    N Z=0:T=39:F=1
470 POKE A+X+Y1,Z:POKE B-X+Y1,Z
    :POKE C+X-Y1,Z:POKE D-X-Y1,
    Z
480 POKE A+Y+X1,Z:POKE B-Y+X1,Z
    :POKE C+Y-X1,Z:POKE D-Y-X1,
    Z
490 Z1=Z:GET A$:IF A$="" THEN 4
    30
500 IF A$=CHR$(13) THEN  RUN220
510 T=0:F=0:GOTO 410
520 END
```

Uno scacciapensieri a zero lire

Un programma che, oltre ad essere un simpatico gioco, è anche simulazione, didattica e suggerimento per un nuovo standard da adottare per imparare a programmare divertendosi

di Alessandro de Simone

Di certo i neo utenti del Commodore 64 si saranno divertiti a digitare il breve programma riportato sul manuale che fa muovere una mongolfiera lungo lo schermo.

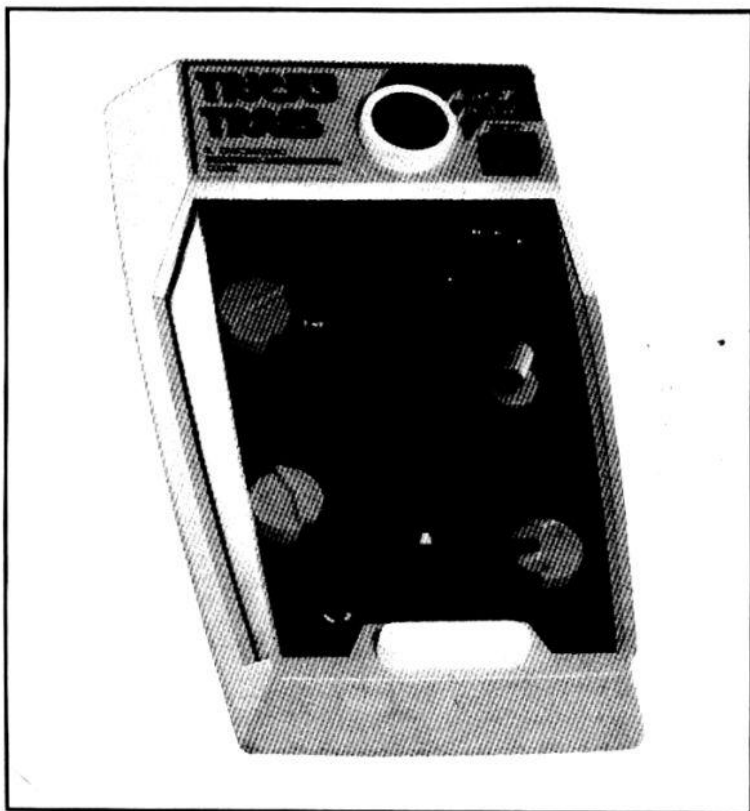
Non tutti, però, pur comprendendo le operazioni elementari necessarie per animare uno sprite, sono in grado di sviluppare giochi per proprio conto e, a maggior ragione, riportarli su computer.

Dalla teoria alla pratica

Il brevissimo programma riportato in queste pagine prende spunto da uno di quei giochi elettronici tascabili che vanno tanto di moda oggi.

Ognuno di questi, tra l'altro, costa una cifra non indifferente (di rado inferiore alle 30000 lire) e non si sottrae, come tutti gli altri, all'antieconomica regola del bel gioco che dura poco...

Perchè, dunque, non gettare le basi per realizzare, con il C/64, questi moderni giocattoli? E perchè, visto che ci siamo, non fare in modo da sviluppare un metodo di programmazione che faciliti la realizzazione di tali giochi?

Si tenga presente che gli "scacciapensieri" (strano nome affibbiato ai

giochini elettronici) basano la loro tecnica su alcune mini-immagini (cristalli liquidi) che vengono accese o spente a seconda del caso; vi sono, inoltre, oggetti o personaggi (anch'essi immagini su cristalli liquidi) da muovere premendo qualche pulsante.

Tale tecnica di animazione sembra fatta apposta per il nostro C/64, che in quanto a sprite (a colori!), a tasti e ad effetti sonori non è certo secondo a nessuno.

Trascurando la banale ironia sulla non tascabilità di un intero sistema C/64, rimane la noia della digitazione dei dati relativi agli sprite. E' noto, infatti, che ognuno di questi richiede ben 63 dati da digitare, quasi sempre, sotto forma di DATA. Molto spesso una digitazione tanto lunga (ben 256 numeri da trascrivere per sole quattro figure) ha scoraggiato più di un lettore dal digitare programmi riportati su riviste del settore.

Lo standard

Non tutti, però, riflettono sul fatto che non è necessario digitare 63 numeri per ciascuno sprite di un gioco che si intende sviluppare, ma è sufficiente assegnare un blocco della memoria che contenga 63 dati non tutti nulli.

In altre parole: se volete realizzare un gioco (come nel caso del programma pubblicato) in cui un'astronave debba esser colpita da un missile, non è necessario disegnare i due oggetti fin dall'inizio. Basterà concentrarsi sullo sviluppo del gioco e far finta che quei rettangoli informi, provvisoriamente assegnati, rappresentino l'uno l'astronave e l'altro il missile.

Quando saremo soddisfatti del nostro gioco (in seguito a vari controlli, modifiche e migliorie) potremo dedicare il nostro tempo a disegnare gli sprite.

Il programma pubblicato

Abbiamo accennato, all'inizio, ad una sorta di Standard per la realizzazione di programmi basati sull'uso degli sprite.

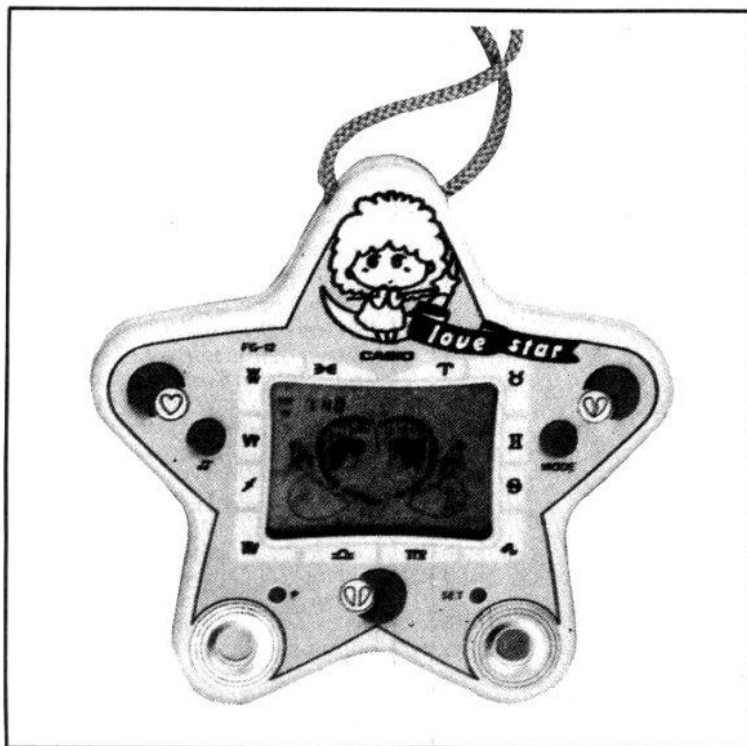
Il vantaggio di una simile adozione consente di digitare il micro-programma (meno di 30 righe di programma, escluse le REM!) che simula uno scacciapensieri.

Si notino le quattro subroutine poste alle righe 400, 500, 600, 700 che assegnano, ai quattro sprite utilizzati, il solo blocco n.0 della memoria. E' ovvio che il primo gruppo di 64 byte contiene informazioni vitali per il computer e non deve in alcun modo essere alterato.

In seguito, se il lettore non sarà soddisfatto del risultato grafico (invero un po' bruttino) potrà digitare le quattro subroutine del secondo listato che, se battute così come sono pubblicate, si "sovrapporranno" al primo listato e sostituiranno il banco n.0 (comune ai quattro sprite) con quelli numerati con 226 (astronave), 227 (cannone), 228 (missile), 229 (bomba). Se lo desiderate potrete digitare il secondo listato anche a... rate; cioè dapprima le righe da 400 a 470, poi quelle da 500 a 570 e così via. E se le immagini non saranno di vostro gradimento potrete sbizzarrirvi a modificarle come preferite, senza modificare di una sola virgola il programma principale (righe 130/350).

Le righe 180-210, opportunamente inserite in qualsiasi listato del genere, vi aiuteranno di certo nella stesura di programmi di vasto respiro che richiedano la gestione degli sprite.

Ai più bravi

Non poteva mancare un invito a coloro che si ritengono più bravi e, magari, anche a coloro che lo stanno diventando: di scacciapensieri, come quello che ha "ispirato" il programma pubblicato, ce ne sono tanti e non sempre vale la pena spendere cifre ragguardevoli per procurarseli. D'altro canto rappresentano una validissima palestra per sperimentare la propria abilità nel programmare: riporatre, su computer, i giochi tascabili è un'operazione semplice solo in apparenza.

Chi se la sente può cimentarsi nella simulazione di qualche giochino in suo possesso ed inviarci il listato, a patto di strutturarlo come indicato (versione semplice e versione "disegnata").

Ai più meritevoli, come al solito, ricchi premi e cotillons.

Prima di inviare il materiale, ma ormai dovreste saperlo, telefonateci: 02/84.67.34.8, meglio se al pomeriggio.

SCHEDA TECNICA

Software applicativo per:

grafica
giochi
didattica
simulazioni

Idoneo per computer:
C/64

Difficilmente adattabile ad altri computer

Ideale l'uso di un monitor a colori

Consigliato a tutti i lettori

```
100 REM SEMPLICI GIOCHI PER PRI
    NCIPIANTI (E NON...)
110 REM SIMULAZIUNE DI SCACCIAP
    ENSIERI PER COMMODORE 64 ME
    DIANTE L'USO DI SPRITE
120 :
130 POKE 56,60: RUN140:REM FISS
    A TOP DI MEMORIA
140 V=53248:PRINTCHR$(147) TAB(
    26)"USARE I TASTI":PRINT TA
    B(26)"CURSORE PER"
150 PRINT TAB(26)"POSIZIONARSI"
    :PRINT TAB(26)"E SPARARE"
160 POKE V+21,255:REM ABILITA T
    UTTI GLI SPRITE
170 P=100:REM INIZIALIZZA VELOC
    ITA'
180 GOSUB 400:REM SPRITE N.0: A
    STRONAVE
190 GOSUB 500:REM SPRITE N.1: C
    ANNONE
200 GOSUB 600:REM SPRITE N.2: M
    ISSILE
210 GOSUB 700:REM SPRITE N.3: B
    OMBA
220 FOR I=V TO V+16:POKE I,0:NE
    XT:REM AZZERA LA POSIZIONE
    DI TUTTI GLI SPRITE
230 W=0:Z=200:K=80:U=30:REM INI
    ZIALIZZAZIONE
240 X=30+INT(RND(0)*4)*60:IF X=
    Y THEN 240:REM  30,90,150,2
    10
250 Y=X:POKE V,X:POKE V+1,50:RE
    M POSIZIONA ASTRONAVE
260 GET A$:REM ESAMINA TASTO PR
    EMUTO
270 IF A$=CHR$(29) AND W=0 THEN
     U=U+60:IF U>210 THEN U=30:
    REM CURSORE IN BASSO
280 IF A$=CHR$(17) THEN W=1:REM
     CURSORE A DESTRA
290 IF W=0 THEN POKE V+2,U:POKE
```

```
     V+3,230:REM  BASE
300 IF W=1 THEN POKE V+4,U:POKE
     V+5,Z:Z=Z-10:IF Z<50 THEN
    Z=200:H=H+1:P=P*.8:GOTO 220
310 POKE V+6,X:POKE V+7,K:K=K+1
    Ø:IF K>230 THEN K=8Ø:H=H+1:
    P=P*.8:GOTO 220:REM  BOMBA
320 FOR J=1 TO P:NEXT:REM RITAR
    DO
330 PRINTCHR$(19)L,H:REM VISUAL
    IZZA BOMBE INTERCETTATE E C
    ANNONI DISTRUTTI
340 IF PEEK(V+30)=252 THEN L=L+
    1:GOTO 220:REM RICOMINCIA S
    E INTERCETTA BOMBA
350 IF H<3 THEN P=P*.95:GOTO 26
    Ø:REM AUMENTA VELOCITA' DEL
     5%
360 END
370 :
380 REM ASSEGNAZIONE SPRITE E C
    OLORI FITTIZI
390 :
400 POKE 2040,Ø:POKE V+39,Ø:RET
    URN
500 POKE 2041,Ø:POKE V+40,1:RET
    URN
600 POKE 2042,Ø:POKE V+41,2:RET
    URN
700 POKE 2043,Ø:POKE V+42,5:RET
    URN
```

```
360 END
370 :
380 :
390 REM ASTRONAVE NEMICA
400 POKE 2040,226:POKE V+39,Ø:F
    OR I=Ø TO 62:READ A:POKE 22
    6*64+I,A:NEXT:RETURN
410 DATA Ø,56,Ø,Ø,255,Ø,7,56,22
    4
420 DATA 7,56,224,7,56,224,7,56
    ,224
430 DATA 7,56,224,7,56,224,7,56
    ,224
440 DATA 7,56,224,7,56,224,7,56
    ,224
450 DATA Ø,255,Ø,Ø,255,Ø,Ø,255,
    Ø
460 DATA Ø,56,Ø,Ø,255,Ø,1,56,12
    8
470 DATA Ø,56,Ø,Ø,255,Ø,1,56,12
    8
480 :
490 REM DATI PER CANNONE
500 POKE 2041,227:POKE V+40,1:F
    OR I=Ø TO 62:READ A:POKE 22
    7*64+I,A:NEXT:RETURN
510 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
520 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
530 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
540 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
550 DATA Ø,126,Ø,Ø,126,Ø,Ø,126,
    Ø
560 DATA Ø,255,Ø,Ø,255,Ø,Ø,255,
    Ø
570 DATA Ø,255,Ø,Ø,255,Ø,Ø,255,
    Ø
580 :
590 REM DATI PER MISSILE
600 POKE 2042,228:POKE V+41,3:F
    OR I=Ø TO 62:READ A:POKE 22
    8*64+I,A:NEXT:RETURN
610 DATA Ø,24,Ø,Ø,24,Ø,Ø,60,Ø
620 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
630 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
640 DATA Ø,126,Ø,Ø,126,Ø,Ø,126,
    Ø
650 DATA Ø,255,Ø,Ø,255,Ø,Ø,255,
    Ø
660 DATA Ø,255,Ø,Ø,255,Ø,Ø,255,
    Ø
670 DATA 3,24,192,7,195,224,7,Ø
    ,224
680 :
690 REM DATI PER BOMBA
700 POKE 2043,229:POKE V+42,5:F
    OR I=Ø TO 62:READ A:POKE 22
    9*64+I,A:NEXT:RETURN
710 DATA 7,195,224,7,Ø,224,3,24
    ,192
720 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
730 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
740 DATA Ø,255,Ø,Ø,255,Ø,Ø,255,
    Ø
750 DATA Ø,126,Ø,Ø,126,Ø,Ø,126,
    Ø
760 DATA Ø,60,Ø,Ø,60,Ø,Ø,60,Ø
770 DATA Ø,60,Ø,Ø,24,Ø,Ø,24,Ø
780 END
```

Un sistema controllato

Come assegnare un voto a ciascuna colonna del vostro sistema Totocalcio

di Alessandro de Simone

Quando sviluppiamo un sistema (sia esso integrale, ridotto o condizionato), otteniamo alla fine un gruppo di colonne da trascrivere sulle schedine per la giocata.

Non tutte le colonne, però, hanno la medesima probabilità di vittoria e ciò vale anche per quelle colonne che derivano da particolari condizionamenti imposti dal giocatore, come nel caso, appunto, dei sistemi ridotti o condizionati.

Quando assegniamo un pronostico fisso ad una partita, riteniamo di essere ragionevolmente certi che si vefifichi l'evento indicato; ne approfittiamo per ricordare che, secondo le leggi della statistica, la "certezza" equivale al 100% delle probabilità che un certo evento si verifichi.

Non sempre, però, quando decidiamo di giocare una "doppia" (come nel caso di un pronostico "1,X") la nostra intenzione è quella di assegnare eguale probabilità alla vittoria e al pareggio (50% per ciascun evento). Può capitare, in casi come quello esaminato, di voler assegnare il 74% alla vittoria in casa e solo il 26% al pareggio (la somma deve comunque essere pari a 100).

Analogamente, nel caso delle triple, è possibile trovarsi nella condizione di voler assegnare probabilità di esito del tutto diverse tra loro: 35% per la vittoria in casa; 55% per il pareggio e solo il rimanente 10% per la vittoria degli ospiti.

Secondo lo stesso ragionamento, attribuire un pronostico fisso (come "1") significa assegnare le seguenti probabilità: "1"=100%; "X"=0%; "2"=0%

Sarebbe interessante, quindi, per ciascuna colonna del sistema sviluppato, determinare il grado di "credibilità" della stessa colonna in base alla probabilità che ogni pronostico ivi ricopre.

Dalla teoria alla pratica

Il programma che pubblichiamo, ovviamente, provvede alla risoluzione del problema in modo semplicissimo.

E' bene sottolineare che il listato agisce su file generati dai programmi "Sistemi condizionati", pubblicato su CCC N.35; "Sistemi Ridotti" (CCC N.37); "Sistemi ridotti e condizionati" (CCC N.38).

Ognuno dei programmi citati elabora un certo numero di colonne che, memorizzate su supporto magnetico secondo uno standard predeterminato, sono suscettibili di lettura e successive elaborazioni da parte dell'utente.

Per illustrare le fasi da percorrere,

in modo da utilizzare correttamente il programma di queste pagine, seguite alla lettera quanto segue ed eviterete difficoltà nelle vostre applicazioni personali.

Preliminari

Supponiamo di voler applicare la determinazione della probabilità al gruppo di colonne generate con il programma "Sistemi ridotti" (CCC N.37). Caricheremo dapprima, naturalmente, questo programma e alla domanda "Inserisci il sistema da ridurre" digiteremo il seguente, formato da due triple, tre doppie e otto fisse:

1: 1
2: 2
3: 1
4: 1
5: 1X2
6: X
7: X
8: 1X
9: X
10: 1
11: 1X
12: X2
13: 1X2

Il programma "Sistemi ridotti", dopo qualche secondo, risponderà indicando il numero di colonne costituenti lo sviluppo integrale (72 colonne), la riduzione praticabile (18 colonne) e la possibilità di svilupparlo; rispondendo "S" alla domanda ne verrà posta un'altra ("Crei archivio?") alla quale risponderemo affermativamente indicando, in seguito, il supporto magnetico desiderato (nastro o disco) ed il nome; questo sarà "TOT" dal momento che il programma di queste pagine accetta solo questo nome (per problemi di compatibilità con il programma "Sistemi condizionati").

Dopo un tempo determinato dalla periferica usata, otterrete sul supporto magnetico il file sequenziale "TOT" pronto per essere manipolato dal programma qui pubblicato.

Come utilizzare il programma

Caricato, quindi, il programma di queste pagine che avrete furbescamente digitato e registrato in precedenza, date il solito RUN.

Verranno poste, per ciascuna delle 13 partite, due domande relative alla percentuale da assegnare ad ognuno dei tre eventi ("1", "2", "X"). Il programma, infatti, non può sapere se le colonne che andrà a leggere dal file sequenziale appartengono, o meno, a pronostici fissi, a doppie oppure a triple.

Poichè alla prima partita abbiamo assegnato in precedenza il pronostico "1" (fisso), evidentemente digitere-

mo 100 alla domanda "Prob.1 (Max: 100)?".

Subito dopo, sempre per la prima partita, il computer chiederà la probabilità da assegnare alla "X". La risposta sarà ovviamente nulla e potremo anche battere il tasto Return senza digitare la risposta: questa, infatti, vale zero per default. La domanda relativa al pronostico "2" non viene posta dal momento che la somma "deve" essere 100 ed il computer, in seguito alla elaborazione delle due risposte precedenti, deduce la terza automaticamente.

Una opportuna domanda di conferma sulle probabilità assegnate ai tre pronostici possibili viene, comunque, effettuata per consentire eventuali correzioni. In caso di risposta affermativa il programma chiederà i dati relativi alle partite successive.

Poichè anche per la seconda, terza e quarta partita abbiamo assegnato pronostici fissi, digiteremo le nostre decisioni assegnando 0 ai pronostici "1" e "X" della seconda partita (con il conseguente 100 assegnato automaticamente a "2"); 100 al pronostico "1" della terza e quarta partita (e 0 sia ad "X" che a "2").

Per la quinta partita si presenta, finalmente, l'opportunità di utilizzare realmente il programma pubblicato.

Supporremo, dunque, di essere più propensi ad una vittoria in casa e ad un pareggio piuttosto che ad una sconfitta. Assegneremo, dunque i seguenti valori:

"1"= 50%
"X"= 35%
"2"= 15%

Senza dilungarci sul comportamento da seguire per le altre "fisse", e non volendo ripeterci per le "variabili", ci limitiamo ad indicare qui di seguito le percentuali ipotizzate nell'esempio:

Partita N.8 (1,X):
"1"= 40
"X"= 60
"2"= 0

Parttita N.11 (1,X):
"1"= 50
"X"= 50
"2"= 0

Partirta N.12 (X,2)
"1"= 0
"X"= 38
"2"= 62

Partita N.13 (1,X,2)
"1"= 35
"X"= 45
"2"= 20

Al termine delle operazioni verrà visualizzato l'intero "quadro" relativo alle percentuali impostate. La solita domanda di conferma farà ripartire il programma oppure consentirà il caricamento del file "TOT" da nastro o disco.

Dopo un periodo di tempo dipendente dalla periferica adoperata e dal numero di colonne elaborate con il programma riduttore, verranno visualizzate tutte le colonne, una per una, corredate da altri dati che ora esamineremo.

La prima colonna indica il numero della partita; la seconda è invece quella da trascrivere eventualmente sulla schedina; l'ultima è relativa alla percentuale impostata per ciascun evento. Vi troveremo, appunto, il 100% nel caso di pronostici fissi, ed altri valori, variabili a seconda delle percentuali assegnate, negli altri casi.

La colonna che merita una spiegazione è la terza ed è proprio quella sulla quale si basa l'intera elaborazione del programma presentato.

Il valore di un milione che compare (vedi riga 260) è un numero ragionevolmente alto (che il lettore dovrà modificare, in più o in meno, a seconda dei sistemi che è abituato a sviluppare) assunto come base per i successivi decrementi. Se una partita ha un pronostico fisso, la cifra rimane inalterata. Se, invece, nella stessa colonna, viene incontrato un pronostico che ha il 70% di probabilità, la cifra stessa viene decrementata nel senso che viene memorizzato il 70% del suo valore.

Nel caso particolare della prima colonna, infatti, la cifra di un milione rimane identica nelle prime quattro partite dal momento che vi corrispondono altrettante "fisse" (=100%). Con il quinto pronostico si ha invece una riduzione del 50% (mezzo milione) che rimane inalterato per la sesta e settima partita.

L'ottava partita riduce la cifra al 40% del suo valore iniziale portandosi, così, a 200000. Procedendo con lo stesso ragionamento si perviene al valore finale 21700 che rappresenta il "voto" assegnato alla colonna N.1.

Naturalmente i valori finali che sono determinati non possono rappresentare un valore assoluto, ma solo relativo a quel particolare sistema elaborato. Continuando a visualizzare i risultati, possiamo annotare i voti attribuiti alle 18 colonne:

1: 21700
2: 27900
3: 12400
4: 19950
5: 25650
6: 11400
7: 15190
8: 19530
9: 8680
10: 13965
11: 17955
12: 7980
13: 6510
14: 8370
15: 3720
16: 5985
17: 7695
18: 3420

Al termine dell'elaborazione viene visualizzato il voto minimo e quello massimo che valgono, nel caso particolare, 3420 e 27900 rispettivamente.

Il programma, dunque, si limita ad indicare un voto in base ad un procedimento matematico basato, tuttavia, su considerazioni personali dell'utente.

A quest'ultimo spetta il compito di stabilire quali colonne giocare selezionando, per esempio, tutte quelle che hanno conseguito un punteggio superiore ad un certo valore, oppure calcolando la media dei voti ("pesata" o meno) da prendere come riferimento per la selezione.

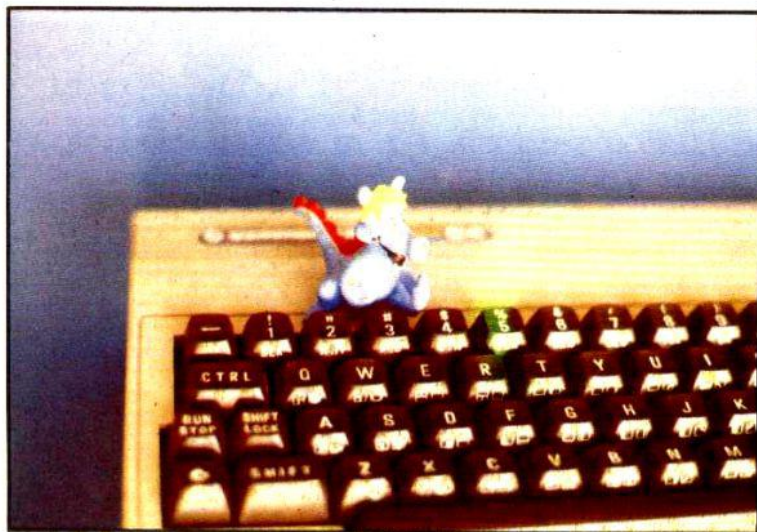
Se il numero di doppie e triple è elevato, è probabile che il valore di un milione sia modesto e generi, alla fine, valori troppo piccoli e scomodi da usare.

E' ovvio che sarà compito del lettore, dopo alcune prove, realizzare quella versione del listato che più risponda alle sue esigenze.

```
100 REM DETERMINAZIONE DELLE CO
    LONNE PIU' PROBABILI ELABOR
    ATE CON I PROGRAMMI
110 REM PUBBLICATI SU COMMODORE
     COMPUTER CLUB
120 REM BY ALESSANDRO DE SIMONE
130 :
140 DIM A$(700):C$=CHR$(147):D$
    =CHR$(17):E$=CHR$(157):F$=C
    HR$(29)
150 FOR I=1 TO 17:G$=G$+CHR$(17
    ):NEXTI:H$=CHR$(19)
160 PRINTC$"PERCENTUALI AL TOTO
    CALCIO"
170 GOSUB 400
180 PRINT:PRINT"I DATI SONO SU
    NASTRO O DISCO? (N/D)":GOSU
    B 540
190 REM "TOT" E' IL NOME STANDA
    RD DEL FILE: MODIFICARLO SE
     NECESSARIO
200 IF A$="N" THEN OPEN 1,1,0,"
    TOT":GOTO 230
210 IF A$="D" THEN OPEN 1,8,12,
    "TOT,S,R":GOTO 230
220 GOTO 200
230 INPUT#1,NR:REM NR=N.COLONNE
     REGISTRATE CON IL PROGRAMM
    A TOT.13 (CCC N.35)
240 PRINTC$"LETTURA DATI"
250 FOR J=1 TO NR:FOR H=1 TO 13
    :INPUT#1,A$:A$(J)=A$(J)+A$:
    NEXTH,J:CLOSE 1
260 MI=1000000:MA=0:FOR J=1 TO
    NR:PE=1000000
270 PRINTCHR$(147)CHR$(18)"COLO
    NNA N."J"  TOTALE="NR
280 FOR K=1 TO 26 STEP 2:W=INT(
    K/2)+1
290 A$=MID$(A$(J),K,1):PE=PE*X(
    W,VAL(A$))/100
300 PRINTW,X$(VAL(A$)),PE,X(W,V
    AL(A$))"%"
310 NEXTK:PRINT:PRINTCHR$(18)"V
    ALORE COLONNA:"PE:IF PE>MA
    THEN MA=PE
320 IF MI>PE THEN MI=PE
330 PRINT"(BATTI P PER ESAMINAR
    E LE PERCENTUALI)"
340 GOSUB 540
350 IF A$="P" THEN PRINTCHR$(14
    7);:FOR I=1 TO 13:GOSUB 560
    :PRINT:NEXTI:GOSUB 520
360 PRINT:NEXTJ
370 PRINT"MINIMO:"MI,"MASSIMO:"
    MA
380 PRINT"VUOI VEDERLI DI NUOVO
    ? (S/N)":GOSUB 540:IF A$="S
    " THEN 260
390 END
400 DIM X(13,3):X$(1)="1":X$(2)
    ="X":X$(3)="2"
410 FOR I=1 TO 3:X$(I)=CHR$(18)
    +X$(I)+CHR$(146):NEXT
420 FOR I=1 TO 13
430 Z=100:FOR X=1 TO 2
440 PRINTCHR$(18)"PART.N."I:PRI
    NT"PROB."X$(X);:PRINT"(MAX:
    "Z")";:INPUT X(I,X)
450 IF X(I,X)>Z OR X(I,X)<0 THE
    N 440
460 Z=Z-X(I,X):IF Z<0 OR Z>100
    THEN 440
470 NEXTX:X(I,3)=Z
480 GOSUB 560:GOSUB 530:IF A$<>
    "S" THEN 430
490 PRINT:NEXTI:GOSUB 570:GOSUB
     530
500 IF A$<>"S" THEN 420
510 RETURN
520 PRINT:PRINT"PREMI UN TASTO"
    :GOTO 540
530 PRINT:PRINT"CONFERMI? (S/N)
    "
540 GET A$:IF A$="" THEN 540
550 RETURN
560 PRINT"PART."I;:FOR X=1 TO 3
    :PRINTX$(X)":"X(I,X);"  ";:
    NEXTX:RETURN
570 PRINTCHR$(147):FOR I=1 TO 1
    3:GOSUB 560:PRINT:NEXTI:RET
    URN
580 END
```

Autorun maker

Una brevissima routine che permette di dotare di autorun un qualsiasi programma Basic

di Michele Maggi

Il problema dell'autorun (caricamento e successiva esecuzione automatica del programma) è già stato abbondantemente trattato sulle pagine della nostra rivista; ciò nonostante l'autorun che proponiamo è, al pari degli altri, notevolmente efficace, ma più semplice e veloce da utilizzare e, perchè no, da studiare in forma disassemblata.

In passato abbiamo visto autorun che presupponevano una modifica al programma da mandare in esecuzione (cfr C.C.C. N.34 "Autorun: variazioni sul tema") o altre procedure che (anche se più efficaci) richiedevano una buona conoscenza del Sistema Operativo da parte dell'utente.

Con la routine proposta, invece, anche l' utente meno informato potrà dotare di autorun i suoi programmi in Basic dal momento che il comando per "trattare" il programma è estremamente semplice:
SYS 828:SAVE"NOMEPRG",8
dove NOMEPRG è il nome del programma che si vuole dotare di autorun.

Come utilizzare Autorun Maker

Una volta digitato il breve programma proposto, sarà opportuno salvarne una copia su disco (PRIMA di impostare il Run) per evitare di doverlo ridigitare in caso di errore che può generare un crash del sistema.

Per utilizzare correttamente la procedura indichiamo qui di seguito le fasi da seguire:

1/ Caricare e mandare in esecuzione il programma Autorun Maker.
2/ Una volta comparsa la scritta "Autorun Maker attivato" è necessario battere il comando New.
3/ Caricare (o digitare) il programma che si vuole dotare di autorun.
4/ Digitare in modo diretto e su una sola linea i seguenti comandi:
SYS 828:SAVE"NOMEPRG",8
facendo attenzione che il nome del programma da proteggere non sia già presente sul disco.

A questo punto inizierà la procedura di salvataggio relativa sia al programma vero e proprio che alla zona di memoria necessaria al funzionamento dell'autorun (da 770 a 2048).

Dato che la suddetta zona di memoria comprende anche la memoria video (1024-2023) le prime istruzioni della routine L.M. provvedono a pulire il video in modo che, con il successivo caricamento, non compaiano sullo schermo le scritte che eventualmente erano presenti al momento del salvataggio.

A salvataggio avvenuto si verificherà un Reset. In seguito, caricando da disco il programma protetto con LOAD"NOMEPRG",8,1 verrà pulito il video e, al termine del caricamento, il programma partirà automaticamente senza possibilità di essere fermato.

Se si tentasse di caricare il programma con il solito LOAD"NOMEPRG",8 si otterrebbero solo simboli e scritte senza senso.

La routine proposta, oltre che a mandare in autorun un qualsiasi programma Basic, lo protegge dal List in quanto ne dirotta i vettori (774-775) alla routine di Reset (64738).

Anche la combinazione dei tasti Run/Stop e Restore viene disabilitata in modo da impedire all'utente di fermare il programma.

Il risultato è più che soddisfacente in termini di efficacia e soprattutto di facilità di utilizzo anche se purtroppo non ne impedisce la copia.

Passiamo ora ad analizzare il funzionamento della routine L.M. per chi non intenda limitarsi ad utiliz-

zarla, ma desideri anche comprenderne il funzionamento.

Come funziona il programma

La routine (per motivi di comodità) è allocata da 828 a 914 ($033C-$0392) nel buffer di cassetta ed è composta da due parti: la prima modifica i vettori di Warm Start e i puntatori di inizio Basic, e la seconda crea l'autorun vero e proprio.

La prima parte, oltre a pulire lo schermo, dirotta i puntatori di Warm Start a $0358, indirizzo di partenza dell'autorun, e disabilita i messaggi "SAVING" ed altri.

Successivamente vengono aggiornati i puntatori di inizio Basic (43 e 44) in modo da farli puntare non più a 2049, ma a 770 sì da includere tutta la zona di memoria da 770 alla fine del programma Basic non appena viene impartito il comando SAVE.

La seconda parte crea l'autorun ricorrendo all'uso del buffer di tastiera; subito dopo resetta il Warm Start e passa a disabilitare List e Run/Stop.

Che cos'è il Warm Start

Sicuramente qualcuno si starà chiedendo che cosa sia il Warm Start e provvediamo subito a soddisfare la sua curiosità.

I puntatori 770/771 ($0302-$0303) indicano al Sistema Operativo (S.O.) l'indirizzo di partenza della cosiddetta routine di Ready che viene chiamata molto spesso.

Si tratta della routine che, oltre a stampare la consueta scritta Ready, provvede a restituire il controllo del cursore e ad altre "cose" ancora.

Ogni volta che il S.O. termina un'operazione di caricamento da periferica, esamina il contenuto dei puntatori di Warm Start perchè intende eseguire il Ready.

Se modifichiamo i puntatori, il S.O. crederà di eseguire il Ready ma, in realtà, eseguirà la routine che parte dall'indirizzo puntato dal suddetto vettore.

Ecco spiegato, dunque, perchè la prima parte della routine provveda al "dirottamento".

Particolare molto importante da ricordare è che una volta eseguita la routine è assolutamente necessario resettare i puntatori ai valori standard e saltare alla routine di Ready.

Studiando il disassemblato ci si potrà sicuramente rendere conto di come sia semplice creare un autorun di buon livello anche con pochi byte.

Disassemblato commentato

```
033c a9 93    lda #$93  ;Pulisce
033e 20 d2 ff jsr $ffd2 ;lo schermo.
0341 a9 58    lda #$58  ;Dirotta i
0343 8d 02 03 sta $0302 ;puntatori
0346 a9 03    lda #$03  ;di Warm Start.
0348 8d 03 03 sta $0303 ;
034b a9 00    lda #$00  ;Disabilita i
034d 85 9d    sta $9d   ;messaggi.

034f a9 02    lda #$02  ;Aggiorna i
0351 85 2b    sta $2b   ;puntatori di
0353 a9 03    lda #$03  ;inizio Basic.
0355 85 2c    sta $2c   ;
0357 60       rts       ;
0358 a9 52    lda #$52  ;Carica 'R' +
035a 8d 77 02 sta $0277 ;Shift 'U' e
035d a9 d5    lda #$d5  ;Return nel
035f 8d 78 02 sta $0278 ;buffer della
0362 a9 0d    lda #$0d  ;tastiera.
0364 8d 79 02 sta $0279 ;Mette 3 nella
0367 a9 03    lda #$03  ;grandezza del
0369 85 c6    sta $c6   ;buffer
036b a9 83    lda #$83  ;Resetta il
036d 8d 02 03 sta $0302 ;puntatore
0370 a9 a4    lda #$a4  ;di Warm Start
0372 8d 03 03 sta $0303 ;
0375 a9 fc    lda #$fc  ;Dirotta il
0377 8d 07 03 sta $0307 ;list.
037a a9 e2    lda #$e2  ;
037c 8d 06 03 sta $0306 ;
037f a9 e1    lda #$e1  ;Disabilita RUN
0381 8d 28 03 sta $0328 ;STOP e RE-
STORE
0384 a5 90    lda $90   ;Se ha salvato
0386 f0 08    beq $0390 ;va a RESET.
0388 a9 93    lda #$93  ;Pulisce lo
038a 20 d2 ff jsr $ffd2 ;schermo.
038d 4c 83 a4 jmp $a483 ;Va a READY.
0390 4c e2 fc jmp $fce2 ;Resetta.
```

```
100 REM AUTORUN MAKER
110 REM C64 & DRIVE
120 :
130 REM BY MICHELE MAGGI
140 :
150 FOR I=828 TO 914
160 READ  A:POKE I,A
170 CK=CK+A
180 NEXT
190 IF CK=9809 THEN 210
200 PRINT"ERRORE NEI DATA":END
210 PRINT"AUTORUN MAKER ATTIVAT
    O!"
220 CLR :PRINT"IMPARTIRE IL COM
    ANDO: NEW"
230 :
240 DATA  169,147,032,210,255,1
    69,088,141,002,003,169,003,
    141,003
250 DATA  003,169,000,133,157,1
    69,002,133,043,169,003,133,
    044,096
260 DATA  169,082,141,119,002,1
    69,213,141,120,002,169,013,
    141,121
270 DATA  002,169,003,133,198,1
    69,131
280 DATA  141,002,003,169,164,1
    41,003,003,169,252,141,007,
    003,169
290 DATA  226,141,006,003,169,2
    25,141,040,003,165,144,240,
    008,169
300 DATA  147,032,210,255,076,1
    31,164,076,226,252
310 END
```

C 16 - PLUS 4

Auto-Run per C/16 & Plus/4

Come impedire ai ficcanaso di esaminare i vostri programmi, anche nel caso in cui ricorrano al vecchio "trucco" del tasto di Reset

di Fabio Calabrò

QQuasi tutti i programmi in commercio, come ben sapete, sono dotati di Auto/Run (A/R), un procedimento in LM che fa automaticamente partire un programma appena caricato in memoria.

Il listato pubblicato in queste pagine agisce nello stesso modo e ne vedremo ora le caratteristiche.

Dovrebbe esser noto che, avendo in memoria un programma Basic, premendo il tasto Run/Stop insieme con il tasto Reset, si entra, come si suol dire, in ambiente Monitor. Senza curarsi di ciò che compare sul video è sufficiente battere i tasti "X" e Return per "rientrare" in Basic. Suggeriremo, ovviamente, anche una procedura per impedire tale "rientro".

Negli indirizzi di memoria $0302 e $0303 sono contenuti, rispettivamente, il byte alto ed il byte basso della locazione di memoria in cui è contenuta l'istruzione da eseguire. Il byte basso ed il byte alto sono rispettivamente le due coppie di caratteri di un indirizzo esadecimale.

I due indirizzi che contengono l'indirizzo di memoria da cui inizia la successiva istruzione da eseguire (in L.M.) vengono comunemente chiamati "puntatori".

Non appena è terminato il caricamento di un programma da cassetta, o da disco, i puntatori sono automaticamente impostati a $12 e $87, vale a dire alla routine di READY, situata all'indirizzo $8712 che provvede a visualizzare la scritta Ready e a far lampeggiare il cursore.

L'A/R fa in modo che, finito il caricamennto, i puntatori siano impostati a $33 e $03, indicando così l'indirizzo $0333, da dove inizia la serie di istruzioni che provvedono all'esecuzione (Run) del programma appena caricato.

Tale gruppo di istruzioni è contenuto nel buffer di tastiera (buffer=area di memoria predisposta per un immagazzinamento temporaneo di dati) situato nelle locazioni da $0527 a 0530; da $0527 a $052A saranno memorizzati i valori $52, $D5, $3A e $0D, che corrispondono all'istruzione RUN (abbreviata nella forma rU) seguita dai due punti e da Return.

Il carattere "R", lo ricordiamo, corrisponde all'esadecimale $52 (decimale 82).

E' ora necessario inserire nella locazione $EF (indice di coda del buffer di tastiera) il valore $04, corrispondente al numero totale di caratteri da considerare all'interno dello stesso buffer.

Dopo che è stato impartito automaticamente il Run al programma, bisogna ripristinare i valori iniziali ($12 e $87) e saltare, subito dopo, alla stessa routine $8712.

Quando si salva (Save) il programma dotato di Autorun su disco o su cassetta, si salva anche la pagina di schermo che inizia da $0C00 e finisce a $0FFF (decimali 3072 4071) e, quindi, tutto ciò che compare sullo schermo al momento del salvataggio. Ciò comporta il fatto che, durante ogni successivo caricamento (Load) del programma dotato di A/R, sullo schermo sarà visualizzato ciò che compariva al momento del salvataggio. Da qui nasce la necessità, prima di salvare, di "pulire" lo schermo.

La protezione è possibile alterando il contenuto del vettore della routine di Save, situato nelle locazioni $0330 (byte basso) e $0331 (byte alto), e trascrivendo $5D $03 per eseguire, prima di salvare, le istruzioni contenute a partire dal byte $035D. Le istruzioni cui accenniamo sono le seguenti:

1/ memorizzare il contenuto dell'accumulatore e dei due registri X e Y nelle locazioni $0390, $0391 e $0392: questi tre registri non sono altro che particolari "contenitori" che permettono di prelevare il contenuto di un byte, manipolarlo per mezzo di operazioni, e di memorizzarlo nuovamente nello stesso, od in un altro, posto; potrebbero essere paragonati alle comuni variabili Basic.

2/ inizializzare l'editor di schermo (cancellare lo schermo)

3/ assegnare la posizione del cursore alla colonna N.0 ed alla 25-esima riga, cioè ad una posizione inesistente.

in modo che il messaggio "Saving..." non compaia su video.

4/ impostare i puntatori del Main indiretto (locazioni $0302 e $0303) a $33 e $03. Il Main indiretto (o ciclo di sistema) è quel gruppo di operazioni che vengono eseguite automaticamente dal computer ogni volta che si finisce di caricare o salvare un programma, oppure subito dopo aver letto la Directory da disco e in altri casi ancora.

5/ ricaricare l'accumulatore ed i due registri con i valori precedentemente memorizzati in $0390, $0391 e $0392.

6/ saltare alla routine di SAVE ($F1A4)

7/ pulire nuovamente lo schermo.

Come proteggere i vostri listati

Abbiamo finora spiegato il funzionamento del breve programma di grande utilità, ma non abbiamo ancora detto come fare per proteggere i programmi Basic da occhi indiscreti.

Seguite alla lettera le operazioni che seguono al termine delle quali ritroverete un file che non sarà facilmente accessibile:

• Impartite il comando New in modo da avere la memoria libera.

• Caricate (Load) il programma in Basic pubblicato in queste pagine ed attivarlo (Run).

• Digitate il programma che intendete proteggere oppure, se disponibile, caricarlo da nastro o disco.

Per ovviare al problema che permette il "trucco" dei tasti Run/Stop e Reset, è indispensabile disabilitare il tasto Run/Stop. Dovete pertanto aggiungere al vostro programma da proteggere la seguente linea:
0 POKE 786,69:POKE 768,26.

• Digitare il comando:
*PRINT HEX$(PEEK(45)+256*PEEK(46))*

Dopo aver premuto il tasto Return, sullo schermo comparirà un numero esadecimale che indicherà l'ultima locazione di memoria occupata dal programma Basic in memoria (vedi CCC n.35 pag.8 "L'ultima spiaggia" per maggiori dettagli) e che dovete annotare a parte su un foglio di carta.

• Entrare in Monitor (digitando "monitor" ⟨R⟩) e scrivere la seguente linea:

⟩ 0330 5D 03

per cambiare il contenuto del vettore Save in modo che, è bene ripeterlo, il computer vada alla locazione $035D per i motivi precedentemente elencati.

E' ora giunto il momento di salvare il nostro programma dotato di autorun e per far ciò digitare (sempre in Monitor) la seguente istruzione:
S"NOMEFILE",A,0302,B

in cui, al posto di "Nomefile" dovremo scrivere il nome che intendiamo dare al programma; al posto di "A" la periferica che si vuole utilizzare (1=registratore, 8=disk-drive) ed al posto di "B" quel numero esadecimale prima annotato.

Premendo il tasto Return lo schermo verrà cancellato mentre il programma sarà inciso su cassetta o su disco; finito il salvataggio, digitando "X" ⟨R⟩ il nostro programma dovrebbe iniziare a girare, prova della perfetta riuscita dell'intera operazione.

Come ha agito il programma

Prima della fase di registrazione, l'autorun provvede ad impostare i puntatori (indirizzi $0302 e $0303) a $33 e $03, in modo che, richiamando il programma, il computer abbia in memoria, oltre all'A/R ed al programma vero e proprio, i puntatori impostati, invece che a $12 e $87, a $33 e $03 e cioè all'indirizzo $0333, dove inizia il programma dell'Autorun.

Come caricare un programma protetto

Per caricare il nostro programma dotato di autorun, si deve digitare in Basic la seguente linea:
LOAD "NOMEFILE ",A,8

valido nel caso del disk drive (per il registratore è sufficiente eliminare ",8) dove per "Nomefile" e "A" valgono le stesse considerazioni di prima.

```
10 rem AUTO-RUN PER C16 & PLU
   S/4
20 rem SOFTWARE BY FABIO CALA
   BRO'- MI -
30 data169,82,141,39,5,169,21
   3,141,40,5,169,58,141,41,5
   ,169,13,141,42,5,234,234
40 data234,169,4,133,239,169,
   18,141,2,3,169,135,141,3,3
   ,32,18,135,96,0,141,144
50 data3,142,145,3,140,146,3,
   32,129,255,24,162,25,160,0
   ,32,240,255,169,51,141,2
60 data3,169,3,141,3,3,173,14
   4,3,174,145,3,172,146,3,32
   ,164,241,32,129,255,96,0
70 scnclr
80 B=0:forI=819to907:readA:
   pokeI,A:B=B+A:nextI
90 ifB<>8878thenprint"ERRORE
   NELLE LINEE DATA!":END
100 new
```

QUALSIASI COMMODORE

Directory N.5

Diecine di programmi e file vari contenuti in centinaia di blocchi; il tutto per una cifra irrisoria

Tutti i programmi contenuti in questo fascicolo sono riportati nel dischetto che proponiamo questo mese. E' inutile dire che, oltre a questi, sono contenuti tanti altri file, utility e programmi che, di solito, decidiamo all'ultimo momento.

Ci soffermeremo, invece, a descrivere il corretto modo di utilizzo di alcuni file presenti in Directory N.4 aggiunti per la gioia di generar sorprese.

Il programma *"Saluti"* è un divertente gadget inviato da un nostro collaboratore; provate a disassemblarlo per capire come funziona.

Il file sequenziale *"CCC/arch/gedafi"* va adoperato con il programma Ge.Da.Fi. pubblicato sul N.37: è un archivio di tutti gli articoli da noi pubblicati a partire dal N.8.

Il programma *duplicatore* (suddiviso in due file) è uno della miriade di utility che circolano liberamente tra gli appassionati del C/64. Spegnete la stampante prima di caricare il primo file con la sintassi ",8,1"; questo caricherà automaticamente la seconda parte del file.

I possessori del *C/16 e Plus 4* saranno certamente felici di trovare, sempre su Directory N.4, ben 12 tra programmi e file da questi generati, oltre a quelli, naturalmente, che figurano tra i listati della rivista.

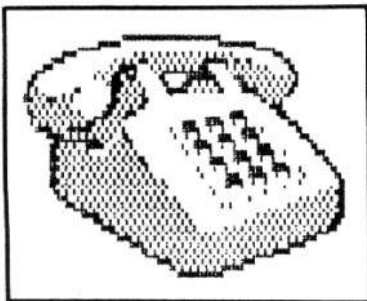
"Fast Format" è un programma brevissimo che consente di formattare ad alta velocità.

"Append" sarà utilissimo a chi si trova nella necessità di fondere due programmi Basic, numerati in successione, presenti su di uno stesso disco.

Altri programmi vari, autoesplicativi, non necessitano di commenti.

Partecipazione dei lettori

Centinaia sono le occasioni per partecipare a "Directory".

Prima di inviare il risultato del vostro lavoro vi consigliamo, però, di telefonarci per stabilire se risponde ai requisiti per l'eventuale pubblicazione (tel. 02/84.67.34.8)

Come procurarsi "Directory"

Avvertiamo i lettori che NON è assolutamente possibile inviare i programmi su nastro, per intuibili motivi di economia ed affidabilità del nastro cassetta.

Ogni numero di "Directory" può quindi esser richiesto SOLO su disco inviando L.12000 per ciascun disco oltre a L.3000 (fisse) per le spese di imballo e spedizione (indipendenti dal numero di dischi richiesti).

Chi desiderasse la spedizione raccomandata, deve aggiungere altre 3000 lire per l'ulteriore affrancatura.

Non ci è possibile inviare materiale contrassegno: si prega di astenersi dal chiedere eccezioni alla regola.

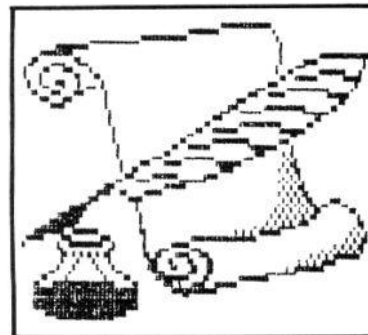
Compilate un normale modulo di C/C postale indirizzando a:

C/C postale N. 37952207
Systems Editoriale
Viale Famagosta, 75
20142 Milano

Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento") non solo il vostro nominativo completo, ma anche il numero del disco desiderato; esempio:

"Directory N.1"
"Directory N.3"
"Directory N.4"

Totale:
L.12000x3
L.6000 (racc.)
= L.42000
(spese di imballo e spediz. racc. comprese).

N.B. Per ottenere il materiale ordinato in tempi più ristretti, inviate l'importo a mezzo assegno bancario non trasferibile con lettera di accompagnamento: le poste italiane non brillano per velocità! (due mesi circa per il recapito di un C/C postale).

LA DIDATTICA E' SYSTEMS

VELOCISSIMO BASIC

per C64/128, C16 & Plus 4, MSX e Spectrum.
Corso completo in 13 lezioni su 4 cassette interamente gestite dal computer. Il corso è diviso in 4 parti, ciascuna delle quali contiene la versione specifica per il computer cui si riferisce

Lire 24.000

ASSEMBLER TUTOR

Un corso completo sull'assembler del C/64 in 8 lezioni interamente gestite dal computer, più un programma *MONITOR*.

Lire 12.000

LOGO 64

La più originale versione del LOGO. Programma non protetto in basic facilmente personalizzabile.

Lire 10.000

24 ORE BASIC

Il corso di basic più veloce per C/64 13 lezioni su 4 cassette con una introduzione "parlata". Tutto il basic senza libri nè dispense.

Lire 24.000

µ PASCAL PER C64

Il volume introduttivo sul Pascal della collana i "libri Systems" completata dalla cassetta con il programma compilatore.

Lire 19.500 (Libro + cassetta)

MS-DOS & GW-BASIC emulator

Il primo programma in grado di emulare sul C/64 il sistema operativo ed il più diffuso basic dei PC Ibm.

Lire 12.000 su cassetta
Lire 25.000 su disco

Si, inviatemi al più presto il seguente software, al prezzo contrassegnato, più lire 3.000 per spese di spedizione:

☐ VELOCISSIMO BASIC (24.000)
☐ ASSEMBLER TUTOR (12.000)
☐ MS-DOS & GW-BASIC EMULATOR
  ☐ versione cassetta (Lire 12.000)
  ☐ versione disco (Lire 25.000)

☐ 24 ORE BASIC (Lire 24.000)
☐ PASCAL PER COMMODORE 64 (Libro +
☐ cassetta lire 19.500)
☐ LOGO 64 (Lire 10.000)

Importo totale lire: ....

Su tale importo mi praticherete lo sconto del 10% in quanto abbonato a ☐ *Commodore Computer Club* ☐ *Personal Computer* ☐ *Computer* ☐ *VR Videoregistrare.* Pertanto vi invio la somma soltanto di lire..........

☐ Desiderando ricevere le copie ordinate con la massima urgenza, accludo assegno bancario n.ro..........
Banca.......... per lire.......... voi intestato.

☐ Contentandomi dei normali tempi postali ho inviato oggi stesso l'importo di lire.......... a mezzo C/C postale N. 37952207 intestato a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Nome ....
via .... **N.ro.** .... **telefono** ....
CAP .... **Città** ....

Come fare una tabella

Utilizzate la stampante in una delle sue principali applicazioni

di Paolo Agostini

HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Se avete comprato da poco una stampante, con queste poche righe Basic potrete non solo ottenere un'utile tabella per convertire in notazione decimale valori esadecimali di 2 cifre (e viceversa) ma, soprattutto, avere a portata di mano un esempio semplicissimo per imparare a realizzare tabelle ordinate di qualsiasi forma e dimensione.

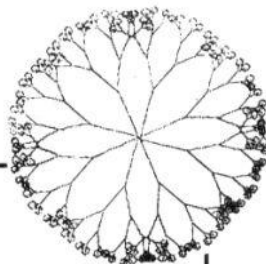
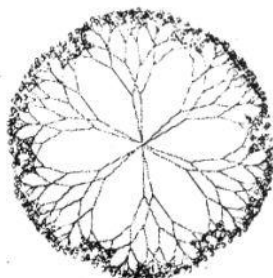
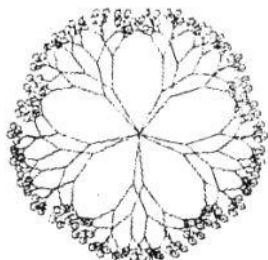
Chi, invece, non possiede una stampante, potrà fotocopiare o ritagliare la tabella dalla rivista. Il suo uso è quasi banale: la prima cifra esadecimale va letta sulla riga verticale, la seconda cifra esadecimale sulla riga orizzontale posta in testa alla tabella.

Per esercizio il lettore provi a realizzare una tabella analoga per la conversione di valori esadecimali a 4 cifre in modo che su stampante sia sempre visualizzata una tabella del formato 16x16, ma relativa ad un gruppo di 256 valori determinati da un Input precedente.

Tanto per chiarire le idee, la tabella qui pubblicata dovrebbe rappresentare l'output con le prime due cifre impostate a "00"; la tabella, ad esempio, impostata con "01" dovrebbe avere come primo valore 256 e, come ultimo, 511, e così via.

Siete sicuri che sia una cosa molto semplice? Provate, e fatecelo sapere...

```
 90 REM TABELLA DI CONVERSIONE
 92 REM DECIMALE - ESADECIMALE
 93 REM BY PAOLO AGOSIINI
100 OPEN 4,4:REM STAMPANTE
110 H$="0123456789ABCDEF"
120 FOR I=1 TO 70
130 PRINT#4,CHR$(61);
140 NEXT:PRINT#4
150 PRINT#4,"HEX";SPC(3);
160 FOR I=0 TO 15
170 PRINT#4,MID$(H$,I+1,1);SPC(
    3);
180 NEXT:PRINT#4
190 FOR I=1 TO 70
200 PRINT#4,CHR$(61);
210 NEXT:PRINT#4
220 FOR I=0 TO 15
230 PRINT#4,CHR$(32);MID$(H$,I+
    1,1);SPC(2);
240 FOR J=0 TO 15
250 V$=STR$(V):V=V+1
260 PRINT#4,SPC(4-LEN(V$));V$;
270 NEXT:PRINT#4:NEXT
280 CLOSE 4:END
```

Curve e simmetrie

Un interessante adattamento di una procedura grafica pubblicata tempo fa

di Domenico Carro

Non ci dilungheremo a descrivere la teoria delle curve di Peano, basate su una procedura matematica particolare (ricorsività) descritta sul N.27 di C.C.C.

Il lettore dovrà limitarsi a digitare il programma e ammirarne le stupende immagini simmetriche che compaiono sul video.

Eventuali errori (in genere: Illegal Quantity) sono dovute a parametri non in regola con i limiti imposti dal computer.

SCHEDA TECNICA

Software applicativo per: grafica

Idoneo per computer: C/16 Plus/4 C/128

Consigliato ai principianti

C/16 Plus/4 C/128

```
100 REM CURVE DI PEANO: PROGRAMMA GRAFICO
110 REM PER COMPUTER C/16, PLUS/4, C/128
120 REM BY DOMENICO CARRO - ROMA
130 :
140 COLOR0,1:COLOR4,1:COLOR1,2,3:SCNCLR
150 PRINT" C U R V E   D I   P E A N O "
160 PRINT"FIGURE MATEMATICHE OTTENUTE SOSTITUENDO PER TANTE";
170 PRINT" VOLTE (GRADO DI DEFINIZIONE),"
180 PRINT"CON LE 'FIGURE DI RACCORDO' PRESCELTE,  I ";
190 PRINT"SEGMENTI COSTITUENTI IL PERIMETRO DI"
200 PRINT"UNA FIGURA REGOLARE. SI OTTENGONO COSI' DELLE ";
210 PRINT"'STELLE' CON UN NUMERO DI BRACCIA"
220 PRINT"PARI AL NUMERO DI LATI DELLA FIGURA"
230 PRINT"  VUOI SCEGLIERE I PARAMETRI ? (S/N)"
240 GETKEYA$:IFA$-"S"THEN310:ELSEIFA$<>"N"THEN240
250 PRINT"PARAMETRI CASUALI O DA PROGRAMMA ?"
260 GETKEYA$:IFA$-"C"THEN610:ELSEIFA$<>"P"THEN260
270 PR-1:K-K+1:IFK>10THENK-1:RESTORE
280 READS,P,N:W-0:GOTO350
290 DATA 4,5,3,6,6,2,5,6,3,4,4,4,5,5,2,5,5
300 DATA 4,4,8,2,3,3,4,6,6,3,4,6,3
310 PRINT""
320 PRINT"NUMERO DI BRACCIA ";:GOSUB590:S-A:PRINTS
330 PRINT"FIGURA DI RACCORDO ";:GOSUB590:P-A:PRINTP
340 PRINT"GRADO DI DEFINIZIONE ";:GOSUB590:N-A:PRINTN
350 IFW-0THENGRAPHIC1,1:W-1:CHAR1,0,0,STR$(S)+STR$(P)+STR$(N)
360 GRAPHIC1:COLOR1,2,3:GR--2*π/P:R--2*π/S
370 FORI-2TOP-1:D(I)--GR:NEXT:D(1)-GR*(P-2)/2:D(P)-D(1)
380 X-160:Y-0:LOCATEX,Y:AN-R/2+π:L-199*SIN(R/2)/3↑N
390 FORI-1TOS:FORK-1TON:O(K)-0:NEXT:GOSUB560:AN-AN+R:NEXT
400 IFPR-1THENFORQ-1TO3000:NEXT:GOTO270
410 IFCA-1THENFORQ-1TO3000:NEXT:GOTO610
420 GETKEYA$
430 GRAPHIC2
440 PRINT"ANCORA ? (S/N)"
450 GETKEYA$
460 IFA$-"N"THENGRAPHIC0:SCNCLR:END
470 PRINT"CANCELLO SCHERMO HI-RES? (S/N)":GETKEYA$
480 W-0:IFA$-"N"THENW-1
490 PRINTCHR$(147):GOTO310
500 K-0
510 K-K+1
520 IFK>NTHENRETURN
530 IFO(K)-PTHENO(K)-0:GOTO510
540 O(K)-O(K)+1
550 AN-AN+D(O(K))
560 X-X+COS(AN)*L:Y-Y-SIN(AN)*L
570 DRAW1TOX,Y:REM ATTENZIONE A NON FINIRE IN "ILLEGAL QUANTITY"
580 GOTO500
590 GETKEYA$:A-VAL(A$):IFA<1THEN590
600 RETURN
610 CA-1:S-INT(RND(0)*3)+4:P-INT(RND(0)*5)+4
620 N-INT(RND(0)*3)+1:W-0:GOTO350
```

I file sequenziali in ambiente Gw-Basic

Una spiegazione dettagliata su come utilizzare la sintassi Gw-Basic attraverso la gestione di un semplice Data Base alla portata di tutti

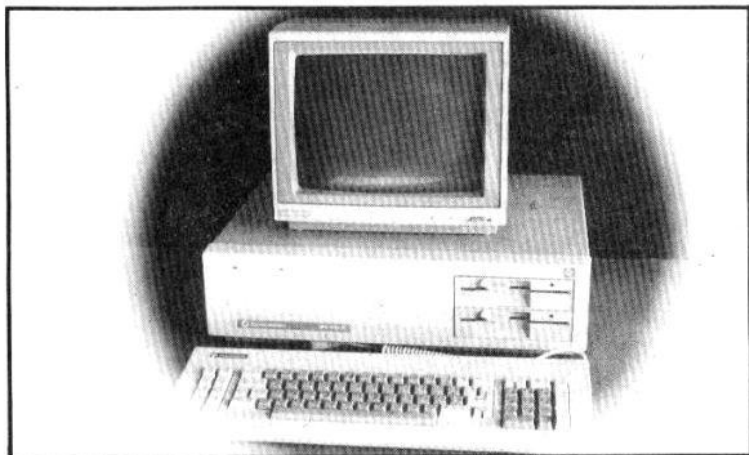
di Roberto Marigo

Come avevamo promesso nel libretto di istruzioni dell'Ms-Dos e Gw-Basic, ci occupiamo, su questa rivista, di programmi da implementare sia sul C/64, dotato di emulatore Gw-Basic, sia su PC-IBM o compatibili.

La sola differenza interesserà l'output su video, che da 40 colonne (caso del C/64) passa a 80 sull'IBM: gli eventuali possessori di quest'ultimo computer potranno, modificando il programma pubblicato, ottenere l'uscita anche sulle 80 colonne: poche Print in meno e qualche Tab in più e il gioco è fatto!

Sottolineamo, comunque, che il programma pubblicato in queste pagine, digitato su un C/64 (dotato di emulatore Gw-Basic) e su un qualsiasi IBM compatibile, fornisce gli stessi risultati.

Il listato, ovviamente, è soprattutto didattico grazie alla semplicità di impostatzione, ma è anche versatile: per una rubrica, una lista di programmi, una serie di dati da riordinare e poi stampare; è, insomma, un piccolo Data Base.

E' possibile naturalmente modificarlo per scopi particolari: chiunque sappia programmare in Basic potrà cimentarsi a modificarlo e a miglioralo con estrema semplicità.

Come usare il programma

All'inizio il programma chiede quanti elementi, al massimo, si desidera inserire. In effetti non c'è un limite vero e proprio se non nella Ram disponibile, ma con un C/64 non si può certo esagerare. Se, infatti, non dovesse comparire subito il messaggio "Out of memory error", questo potrebbe esser visualizzato nel momento meno opportuno, per cui sarà bene salvare l'archivio ad intervalli regolari, per evitare la perdita di dati (e di tempo); ricordate che il C/64 dispone, in ambiente Gw-Basic, di soli 26K-Ram, e il PC-IBM di circa 64K, in cui deve essere allocato anche il programma e le altre variabili che, da soli, occupano 5-6K circa.

Dopo il numero di elementi da memorizzare, verrà richiesto il numero di campi per ciascun elemento e se si desidera, o meno, assegnare una lunghezza prefissata ad ogni campo. In caso di risposta positiva (premendo il tasto "S") il computer ne chiederà la lunghezza massima, altrimenti questa assumerà il valore di default 40; ciò significa che ogni elemento potrà al massimo assere lungo 40 caratteri. Qualora si desiderasse modificare il valore di default, si dovrà cambiare il valore 40 in linea 190.

In ogni caso la lunghezza massima di una stringa sarà di 255 caratteri.

Completate le operazioni descritte, il computer presenterà un menu di opzioni da selezionare premendo il tasto corrispondente.

1-Inserimento Dati

Viene visualizzato il numero di elementi liberi e quello relativo al record che ora inserirete; il computer chiede, campo per campo, l'elemento corrispondente; se il primo elemento è nullo, il computer tornerà al menu senza proseguire nel lavoro. Inseriti i dati il computer chiede conferma circa la loro validità.

2-Ordinamento dati

Con questa opzione il computer chiede conferma in modo da evitare un ordinamento indesiderato. Premendo "S" si accederà all'ordinamento vero e proprio in base al campo specificato.

3-Visualizza Dati

Viene visualizzato, elemento per elemento, l'intero archivio: premendo la barra spaziatrice si proseguirà nella visualizzazione dell'elemento successivo, fino al termine dell'archivio.

4-Stampa Dati

Il computer stampa sulla stampante, provvidenzialmente accesa in precedenza, l'archivio memorizzato.

5-Save Dati

Con questa funzione (previa conferma) il programma salverà su disco l'archivio presente in memoria dotandolo del nome assegnato che deve essere lungo al massimo otto caratteri (in accordo con i limiti del "vero" Gw-Basic), a cui suggeriamo di aggiungere il suffisso ".dat", che corrisponde alla estensione per i computer Ms-Dos.

Se si inserisce, al posto del nome, il simbolo del dollaro ($), il computer mostrerà la directory del disco per poi chiedere nuovamente il nome da assegnare al file. Se battete Return "a vuoto" tornerete al menu.

6-Caricamento dati

Dopo l'abituale conferma, il computer chiederà il nome del file da caricare. Anche in questo caso si ha la possibilità di visualizzare la directory del dischetto o di tornare al menu. Ricordate però che l'archivio salvato deve essere delle stesse dimensioni (elementi e campi) che sono state inserite all'inizio, per evitare spiacevoli conseguenze. Un consiglio: scrivete il numero degli elementi e dei campi nel nome stesso del file al momento di salvarli, per poterli poi ricordare più facilmente.

7-Fine Programma

Dopo la necessaria conferma si uscirà dal programma perdendo i dati in memoria: attenzione ad usarlo saggiamente.

Come funziona il programma

Le note che seguono potranno esser comprese soprattutto dagli esperti ma, se lette con attenzione, saranno comprensibili anche per coloro che sono alle prime armi in fatto di programmazione.

Linee 100-230: Inizializzazione

Viene richiesto il numero degli elementi, quello dei campi e le loro lunghezze. Il tutto è di fondamentale importanza per il programma.

Linee 240-380: Menu

Serve per le scelte, presentate con semplici comandi Print. Ecco alcune note circa le modifiche da apportare per ottenere l'output su video dei compatibili IBM: aumentando i valori delle Tab si potrà centrare sullo schermo il menu, che altrimenti risulta spostato sulla sinistra; per il C/64 il Menu è perfettamente centrato.

Linee 430-560: Inserimento dati

E' la funzione principale del programma: in queste poche righe vi sono le richieste circa i dati memorizzati nella matrice DT$, e la regolazione della lunghezza della stringhe, il cui valore si trova nella matrice LN%

Linee 570-740: Ordinamento Dati
La funzione di bubble sort qui descritta è una delle più veloci e sicure: le stringhe vengono confrontate una ad una e poi riordinate; la procedura è valida per qualunque computer dotato di una qualsiasi versione Basic.
Linee 750-880: Visualizza dati
Viene presentato, con due semplici cicli nidificati, l'intero archivio, elemento per elemento; la pressione di un tasto consente la stampa dell'elemento successivo.

Linee 890-980: Stampa Dati
Con gli stessi due cicli nidificati viene stampato l'archivio.

Linee 990-1260: Save e load dati
Le due routine sono simili (e, soprattutto, simmetriche) e servono per caricare e salvare i file sequeziali di dati. Possono essere migliorate inserendo maggiori controlli sul disco (controllo errore, comandi al Dos), ma in questo caso è necessario effettuare personalizzazioni in base al computer posseduto.

Conclusione

Ribadiamo che il programma qui presentato è di uso soprattutto didattico, ma anche pratico: invece di programmi specifici per rubriche, liste ed altro, il listato può essere una valida alternativa sia per la brevità che per la versatitilità che lo caratterizza.
Un'ultima nota: il C/64 ha due set di caratteri, l'IBM no. Con il C/64 si può passare da uno all'altro con facilità (premendo i tasti Commodore e Shift), ma il programma stampa solo nel set Maiuscolo/Grafico ed è preferibile adottare questo nei propri archivi: un'eventuale modifica è possibile, LPrintando su stampante il comando relativo.
Infine nel simulatore Gw-Basic del C/64 il comando INKEY$ ha lo spiacevole effetto di lasciare intatto il buffer di tastiera con la conseguenza che ad un successivo comando di Input, appariranno i tasti premuti in precedenza, richiesti dall'istruzione INKEY$ (scelta del menu, delle conferme, eccetera). E' sufficiente però battere le seguenti linee per ottenere un output più "accettabile":

165 POKE198,0
355 POKE198,0
525 POKE198,0
585 POKE198,0
765 POKE198,0
855 POKE198,0
905 POKE198,0
1005 POKE198,0
1145 POKE198,0

Il programma, così come si presenta, è tuttavia perfettamente funzionante. La modifica sopra descritta è valida per il solo C/64 e non riguarda le macchine IBM compatibili, su cui produrrebbe risultati imprevedibili.

```
10 rem gestione dei file sequenziali
20 rem in ambiente "Gw-Basic"
30 :
40 rem programma idoneo per computer "Ms-Dos" compatibili
50 rem" e Commodre 64 dotato di emulatore Gw-Basic
60 rem" della Systems Editoriale
70 :
100 rem ************* inizializzazione
110 cls : print : print : print tab(5) "Quanti elementi max " ;
120 input em
130 print tab(5) "Quanti campi per elemento" ;: input cc
140 dim dt$ ( em, cc ) : in = 1
150 print "Vuoi dare una lunghezza per ogni campo   (s/n) ?" :
160 x$="" : x$ = inkey$ : if x$ <> "s" and x$ <> "n" then 160
170 dim ln% (cc)
180 if x$="s" then 200
190 for i = 1 to cc : ln% (i) = 40 :next i : goto240
200 for i = 1 to cc
210 ln%=0:print "Lungh. campo "i ;: input ln%
220 if ln%>255 or ln%=<0 then 210 : else ln%(i)=ln%
230 next i
240 rem ********************* menu
250 cls : print : print : print : print tab(13) "Menu Principale"
260 print tab(13) "---------------"
270 print : print : print tab(10) "1 - Inserimento Dati"
280 print tab(10) "2 - Ordinamento Dati"
290 print tab(10) "3 - Visualizza Dati"
300 print tab(10) "4 - Stampa Dati"
```

```
310 print tab(10) "5 - Save Dati"
320 print tab(10) "6 - Caricamento Dati"
330 print tab(10) "7 - Fine Programma"
340 print : print : print tab(13) "Quale scelta ?"
350 x$="" : x$= inkey$ : if x$ = "" then goto 350
360 if val(x$) <1 or val(x$) >7 then x$="" : goto 350
370 on val(x$) gosub 430,570,750,890,990,1130,390
380 goto 240
390 rem **************** fine programma
400 cls : print : print : print
405 print : print : print : print "Sei sicuro (s/n)?"
410 x$="" : x$ = inkey$
415 ifx$ = "n" then return: else if x$ <> "s" then 410
420 end
430 rem ************** inserimento dati
440 if em - in <0 then return
450 cls : print : print "Elementi Rimanenti :" em-in
460 print "Elemento Numero " in
470 for i = 1 to cc
480 print : print : print "Campo n." i ;
490 cc$="" : input cc$ : if cc$ = "" and i=1 then return
500 cc$ = cc$+space$(ln%(i)) : dt$ ( in, i) = left$(cc$,ln%(i))
510 next i
520 print : print : print "Tutto Ok (s/n) " ;
530 x$ ="" : x$ = inkey$ : if x$ = "" then 530
540 if x$ = "s" then in = in +1 : return
550 if x$ = "n" then return
560 goto 530
570 rem ******************* ordinamento
580 cls : print : print : print : print "Tutto ok (s/n) ?" ;
590 x$ ="" : x$ = inkey$ : if x$ = "" then 590
600 if x$ = "n" then return
610 print : print : input "Ordinamento su quale campo "; co
620 if co <0 or co > cc then 610
630 ix=in-1
640 od = 0 : rem inizio del bubble sort
650 for i = 1 to in-2
660 if dt$(i,co) > dt$(i+1,co) then gosub 710
670 next i
680 ix =ix-1
690 if (ix>1) and (od=1) then 640
700 return
710 for ii = 1 to cc : x$ = ""
720 x$ = dt$(i,ii) : dt$(i,ii) = dt$(i+1,ii) : dt$(i+1,ii) = x$
730 next ii
740 od = 1 : return
750 rem ************** visualizza dati
760 cls : print : print : print"Tutto ok (s/n) " ;
770 x$="" : x$= inkey$ : if x$ = "" then goto 770
780 if x$ = "n" then return
```

Il grande software made-in-Italy

LA VOCE III

Fa parlare e cantare il C64 secondo come lo programmi senza l'uso di campionatori nè sintetizzatori.
Tutte le parole o le canzoni così prodotte possono essere inserite come stringhe in altri programmi.

Lire 12.000

RAFFAELLO

Un programma per disegnare col tuo Comodore 64 col solo joystick senza Koala nè tavoletta grafica.
Tutti i disegni prodotti possono essere memorizzati ed utilizzati in altri programmi.

Lire 10.000

OROSCOPO

Fa in maniera scientifica l'oroscopo personale. Il più completo programma astrologico per Commodore 64.

Lire 12.000

COMPUTER MUSIC

Un music-editor avanzato più per un programma juke-box con 27 motivi celebri di musica classica e leggera da Arcadia a Bach, Vivaldi, Zeppelin...

Lire 12.000

GESTIONE FAMILIARE

Tre programmi su cassetta che giustificano l'aggettivo "domestico" del tuo computer:
- bilancio familiare;
- dieta equilibrata;
- scheda medica familiare.

Gira su C/64/128

Lire 12.000

BANCA DATI

Un potente data base per C/64 e Spectrum disponibile anche su disco.
L'edizione su cassetta contiene da un lato la versione C64 e dall'altro la versione Spectrum.

Lire 12.000

DICHIARAZIONE DEI REDDITI (740/S)

Programma aggiornato al 1986 per la dichiarazione dei redditi, modello semplificato. Per C64.

Disco: **Lire 20.000**
Cassetta: **Lire 12.000**

MATEMATICA FINANZIARIA

Pubblicato a puntate su Commodore (n.ri 13, 14 e 15) e su Personal Computer (n.ri 1, 2, 3 e 4) questo programma offre un vero e proprio corso completo di ragioneria su Commodore 64.
Se ne consiglia l'acquisto insieme agli arretrati delle riviste che ne illustrano l'uso ed il funzionamento.

Disco: **lire 20.000**
Cassetta: **lire 10.000**

Comodore 13, 14 e 15 e Personal Computer 1, 2, 3 e 4
Lire 21.000

ANALISI DI BILANCIO

Naturale completamento di "Matematica Finanziaria" questo programma consente di calcolare automaticamente tutti i ratio più significativi e di confrontare due bilanci dello stesso ente.
Il testo esplicativo è stato pubblicato su Personal Computer n.ri 2, 3, 4 e 5 che si consiglia di acquistare contemporaneamente.

Disco: **Lire 20.000**
Cassetta: **Lire 10.000**
Personal Computer 2, 3, 4 e 5: **Lire 12.000**

ARREDARE

Un programma professionale per ottimizzare le soluzioni d'arredamento della vostra casa.
N.B. gira solo sotto Simon's Basic.

Disco: **Lire 20.000**
Cassetta: **Lire 10.000**

GRAPHIC EXPANDER 128

Un potente programma grafico per il c 128 in modo 128.

Lire 20.000

Sì, inviatemi i seguenti software al prezzo di listino + Lire 3.000 per spese di spedizione:

☐ RAFFAELLO
☐ LA VOCE III
☐ OROSCOPO
☐ COMPUTER MUSIC
☐ GESTIONE FAMILIARE
☐ BANCA DATI
☐ MATEMATICA FINANZIARIA/DISCO
☐ MATEMATICA FINANZIARIA/CASS.
☐ MATEMATICA FINANZIARIA/RIVISTE
☐ ANALISI DI BILANCIO/DISCO
☐ ANALISI DI BILANCIO/CASS.
☐ ANALISI DI BILANCIO/RIVISTE
☐ DICHIARAZIONE DEI REDDITI/DISCO
☐ DICHIARAZIONE DEI REDDITI/CASS.
☐ ARREDARE/DISCO
☐ ARREDARE/CASSETTA
☐ GRAPHIC EXPANDER/DISCO

Valore complessivo: Lire..

Su tale importo mi praticherete lo sconto del 10% in quanto abbonato a ☐ *Commodore Computer Club* ☐ *Personal Computer* ☐ *Computer* ☐ *VR Videoregistrare.* Pertanto vi invio la somma soltanto di lire..

☐ Desiderando ricevere le copie ordinate con la massima urgenza, accludo assegno bancario n.ro..................................
Banca.. per lire.. voi intestato.

☐ Contentandomi dei normali tempi postali ho inviato oggi stesso l'importo di lire......................... a mezzo C/C postale N. 37952207 intestato a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Nome ....
via .... **N.ro.** .... **telefono** ....
CAP .... **Città** ....

```
790 if x$ <>"s" then 770
800 for i = 1 to in-1
810 print "Elemento Num. "i" :"
820 for ii = 1 to cc
830 print dt$(i,ii)
840 next ii
850 print "-----------------------------------------"
860 x$="" : x$= inkey$ : if x$ <> " " then goto 860
870 next i
880 return
890 rem ******************** stampa dati
900 cls : print : print : print"Tutto ok (s/n) " ;
910 x$="" : x$= inkey$ : if x$ = "" then goto 910
920 if x$ = "n" then return
930 if x$ <>"s" then 910
940 for i = 1 to in-1
950 for ii = 1 to cc
960 lprint dt$(i,ii) " ";
970 next ii : lprint : next i
980 return
990 rem ********************** save dati
1000 cls : print : print : print"Tutto ok (s/n) " ;
1010 x$="" : x$= inkey$ : if x$ = "" then goto 1010
1020 if x$ = "n" then return
1030 if x$ <>"s" then 1010
1040 no$="":print : print : print : input "Nome File di dati"; no$
1050 if no$="" then return
1060 if no$ ="$" then files : goto 1040
1070 open "o", #1, left$ (no$,8)+".dat"
1080 for i = 1 to in-1
1090 for ii = 1 to cc
1100 print#1,dt$(i,ii)
1110 next ii : next i
1120 close 1 : return
1130 rem ************** caricamento dati
1140 cls : print : print : print"Tutto ok (s/n) " ;
1150 x$="" : x$= inkey$ : if x$ = "" then goto 1150
1160 if x$ = "n" then return
1170 if x$ <>"s" then 1150
1180 no$="":print : print : print : input "Nome File di dati"; no$
1190 if no$ ="$" then files : goto 1180
1200 if no$="" then return
1210 open "i", #1, left$ (no$,8)+".dat"
1220 for i = 1 to em
1230 for ii = 1 to cc
1240 input#1, dt$(i,ii)
1250 nextii
1260 if eof(1)=1 then goto 1280
1270 nexti
1280 in=i+1:close1 : return
```

Software Club

C64/C128

Cover (12 K)
Target (35 K)
Video puzzle (30 K)
Kiss me (30 K)
Cosmic Ark (35 K)
Speedway (30 K)
Autorun (2 K)

Vc 20

Batbusters (3 K)
China Game (2 K)

C16/+4

Cover (3 K)
Batbursters (3 K)
Rescue (9 K)
Turbo Tape (4 K)

Spectrum

Village III (11 K)
Visitors IV (16 K)
CW-OM (15 K)
Oscar 10 (13 K)

MSX

Turbo (9 K)
Karate III (20 K)
Sort Directory (6 K)

Systems

14 Lire 8.000
Commodore Club - Dir. Resp. A. Ronchetti Edizioni Systems Editoriale Srl - V.le Famagosta 75 -20142 Milano - Reg. Trib. MI. n. 104 del 25/2/84 - Distr. MePe.

Software Club #14

Ecco i programmi di Software Club #14

E' in edicola il nuovo numero di Software Club che, come sempre, contiene numerosi programmi per i computer più diffusi: C/64-128, C/16, Vic 20, Spectrum e MSX.
Riportiamo qui di seguito i "temi" relativi ai programmi dedicati agli utenti Commodore.

COMMODORE 64

Target

Una favolosa simulazione di poligono di tiro. Allenatevi a centrare i bersagli posti davanti a voi con le vostre armi, il fucile ad aria compressa, la pistola calibro 22 e quella calibro 38.

Caricato e lanciato il programma, il computer chiede il numero dei giocatori e il loro nome.
Conclusa questa fase potete scegliere a quale esercitazione partecipare: la vostra arma è controllata dal Joystick in porta 2.

Il computer presenta le tre possibili prove: il tiro col fucile ad aria compressa, la pistola calibro 22 o 38; inserendo il numero relativo alla prova prescelta e premendo RETURN si entra nel poligono ...

Fucile ad aria compressa:

Avete a disposizione 10 colpi per colpire il bersaglio, una serie di quadrati concentrici, e più vi avvicinerete al centro più punti vi verranno attribuiti; in alto viene visualizzato il

nome del giocatore, i colpi a disposizione, i punti relativi al tiro effettuato e i punti totalizzati fino a quel momento.

Ogni volta che sparerete (premendo il tasto FIRE) il computer vi indicherà dove avete sparato con un punto lampeggiante; premendo di nuovo il fire potrete proseguire nella prova.

Pistola calibro 22

Con 5 colpi a disposizione dovete abbattere una serie di sagome che avete dinanzi a voi in un tempo limitato; attenti a non sprecare colpi, perchè dopo due sbagli sarete esclusi e la vostra prova sarà finita.

Pistola calibro 38

Entro un deteminato tempo dovete colpire le sagome che appariranno una per volta in rapida successione. Ma il calibro 38 non è poi così facile come sembra. Avete solo 6 colpi per totalizzare più punti possibile.

Il controllo dell'arma è molto sensibile, per cui usate il joystick con cautela.

Videopuzzle

Un appassionante rompicapo che vi farà passare ore al video compromettendo la vostra salute mentale... Una vera manna per gli appassionati di puzzle.

Partito il programma verrà visualizzata una schermata che potrete osservare solo per pochi secondi.

Ora il programma vi chiederà il livello di difficoltà, espresso in due paramentri, uno per la divisione orizzontale e uno per quella verticale: potrete inserire un valore, indicante il numero di pezzi, da 2 a 12 nella divisone orizzontale e da 2 a 20 in quella verticale.

Dovrete ricostruire la schermata scambiando i vari pezzi: premendo Fire su uno di essi e quindi su di un altro otterrete lo scambio di posizione tra i due.

Completato il puzzle il programma vi indicherà il tempo impiegato e il livello di difficoltà, in base ai quali calcolerà il punteggio.

Durante la fase di soluzione del puzzle, premendo la barra spaziatrice, sarà possibile visualizzare per qualche secondo la figura originale.

Allenate la vostra mente a ricordare i più piccoli particolari del disegno e a ricomporlo nel minor tempo possibile!

Kiss Me

Questa volta avete superato ogni limite: almeno potevate risparmiare le donne sposate! Ora i loro mariti gelosi vi inseguono desiderosi di vendetta.

Dovete difendervi da loro onde evitare di finire K.O. a causa di una delle loro randellate.

Superata una nuova fase potrete raggiungere finalmente la casa di un'altra conquista che vi attende trepidante!

Buon divertimento!

Cosmic Ark

Siete ora al comando di un'arca spaziale e dovete superare le barriere di asteroidi che si frappongono tra voi e il pianeta sul quale sono prigionieri i vostri uomini.

Una volta entrati in orbita e sganciato il modulo di atterraggio dovete superare le stazioni di difesa del pianeta e raccogliere i vostri compagni che vivono prigionieri sul pianeta.

Attenti, la superficie del pianeta è pericolosa, dovrete quindi prendere gli amici senza toccare il suolo.

Il gioco è controllato dal joystick in porta due.

Speedway

Una entusiasmante corsa che vi vedrà protagonisti alla guida della vostra moto.

Dopo aver scelto il livello iniziale di difficoltà e il numero di giri della pista potrete sfidare gli avversari su una piccola pista circolare le cui curve devono essere affrontate in continue sbandate per arrivare al traguardo nel minor tempo possibile.

Il joystick in porta 2 controlla la posizione della moto.

Autorun maker

Come tutte le cassette di Software Club, anche questa è corredata di una fantastica utility: Autorun maker.

Una volta caricato in memoria, il programa può essere attivato con 'SYS 52480'.

Da questo momento in poi avrete a disposizione, oltre agli usuali comandi Commodore, un'istruzione nuova e potente:

S "nomeprogramma"

Questa istruzione permette di proteggere i vostri programmi salvando su nastro (e non su disco) il programma BASIC residente in memoria in modo che, all'atto del caricamento, questo parta automaticamente e non sia più possibile apportarvi alcuna modifica.

Oltre a questa protezione c'è da sottolineare che il programma così salvato non può essere interrotto neppure con RUN/STOP + RESTORE.

La nuova procedura di salvataggio struttura il programma di partenza in due files complementari inscindibili uno dall'altro: tale caratteristica assicura la protezione.

La procedura corretta per l'utilizzo di questo programma è la seguente:

1- caricare e attivare il programma Autorun maker. (SYS 52480)
2- caricare in memoria il programma Basic che si intende proteggere.
3- digitare il nuovo comando S"NOMEFILE"

A questo punto la procedura è terminata, il programma salvato con Autorun maker andrà automaticamente in esecuzione senza possibilità di essere fermato.

COMMODORE C16/Plus4

Batbusters

Questo bellissimo gioco per C16 costituirà un entusiasmante divertimento per svariate ore.

Il gioco consiste nell'abbattere tutti i pipistrelli che incontrerete durante il vostro viaggio facendo attenzione a non urtare tutti gli ostacoli in cui vi imbatterete.

Con i tasti che verranno visualizzati sullo schermo potrete muovervi su e giù , oppure accelerare o sparare.

Ma fate attenzione!

Rescue

Ecco un altro videogame ispirato ad un gioco da bar.

Vi trovate a bordo di un elicottero e la missione è salvare i soldati prigionieri in una zona nemica. Dovete atterrare e caricarli a bordo, evitando di entrare in collisione con le postazioni avversarie e di farvi abbattere dai missili che vi vengono lanciati.

Il gioco viene controllato dalla tastiera.

Turbo Tape

Finalmente anche per voi, utenti C16, è arrivato il programma che ridurrà notevolmente le interminabili attese davanti al vostro registratore.

COMMODORE VIC 20

Batbusters

Vedi istruzioni C/16.

Chinagame

Ora anche sul Vic 20 avete a disposizione un rompicapo che speriamo vi intratterrà per svariate ore.

Il gioco altro non è che la trasposione su computer dell'antico gioco della dama cinese.

Attivato il programma vi si presenterà una scacchiera esagonale con un certo numero di pedine ed un buco al centro.

Lo scopo del gioco consiste nel mangiare tutte le pedine (come nella dama) e di lasciarne esclusivamente una al centro.

Per muovervi lungo la scacchiera dovrete usare i tasti di cursore, quindi per selezionare le pedine e la relativa posizione di arrivo utilizzate la barra spaziatrice.

A tutto disco.

Finalmente, viste le numerose richieste, d'ora in poi ogni pubblicazione ***Software Club*** *su cassetta sarà disponibile anche su dischetto da richiedersi, per corrispondenza, presso la redazione.*

Sono disponibili i seguenti titoli:

Software Club #11 *(C/64-128, C/16, Plus/4 e Vic 20)*
Software Club #12 *(C/64-128, C/16, Plus/4 e Vic 20)*
Software Club #13 *(C/64-128, C/16, Plus/4 e Vic 20)*
Software Club #14 *(C/64-128, C/16, Plus/4 e Vic 20)*

I Gialli Commodore *(C/64-128)*

Charlie Deus *(C/64-128)*

La voce III *(C/64-128)*

Il prezzo, per ognuna delle suddette pubblicazioni è di L. 12.000 più L. 3.000 per spese di spedizione.

Le richieste vanno indirizzate a:

Systems Editoriale
Viale Famagosta, 75
20142 MILANO
Tel. 02/8467348

Il pagamento può essere effettuato tramite assegno bancario o versamento sul c/c N. 37952207
Non è possibile inviare materiale in contrassegno nè contro invio di vaglia telegrafico.
Per ogni ordine, anche se per più dischetti, le spese rimangono fissate in L. 3.000

Sempre un passo avanti.

Voltiamo pagina

Un discorso sullo Stack Pointer, con riferimenti al Basic

di Claudio Baiocchi

Nei Computer Commodore la "pagina 1" (locazioni di memoria numerate da 256 a 511) è, tra l'altro, sede dello "stack". In questo articolo vedremo alcuni dei comandi per i quali lo stack svolge un ruolo determinante. Qua e là affioreranno inevitabilmente riferimenti ad importanti locazioni in pagina 0; il lettore che voglia seguire in tutti i dettagli quanto detto in questo articolo può rileggere gli articoli "Due memorie molto strane", CCC N.29 per i riferimenti alle locazioni 57 e 58. Anche l'articolo "Un'occhiata all'esecutore ed una all'editore", CCC N.35 sarà utile per ottenere maggiori informazioni sulle locazioni 61, 62, 122 e 123.

Chi non ha la fortuna (!) di possedere i due fascicoli citati può, comunque, seguire egualmente questo articolo: sarà sufficiente decidere di "studiarlo" e non semplicemente di "leggerlo".

La Grande Libreria Systems

Autori Vari

64 Programmi per Commodore 64

Giochi, grafica, gestione delle stringhe, musica, numeri, gestionali.

Lire 4.800

Autori Vari

I miei amici C16 & Plus4

Un manuale pratico per padroneggiare il basic di questi computer.

Lire 7.000

Autori Vari

Strategie vincenti per Commodore 64

Le strategie per tutti i classici del videogioco: per giocarli, vincerli o programmarli.

Lire 5.800

Autori Vari

62 Programmi per il Vic 20, C16 e Plus 4

Giochi, grafica e routine per imparare a programmare.

6.500

Roberto Didoni, Guido Grassi

Utilities e giochi didattici

Raccolta di programmi pratici per tutti i Commodore e lo Spectrum.

Lire 6.500

Giovanni Mellina

Tutti i segreti dello Spectrum

4 passi nella Rom: come usare le più importanti routine del sistema operativo.

Lire 7.000

Roberto Didoni, Guido Grassi

Simulazioni e test per la didattica

Teoria e listati per Vic 20, C16, C64 C128 e Spectrum Sinclair.

Lire 7.000

Paolo Goglio

Impara giocando il basic dello Spectrum

Esercizi pratici per entrare nel vivo della programmazione.

Lire 7.000

Clizio Merli

µPascal per Commodore 64/128

Un manuale completo per il programma compilatore

Lire 7.000

Umberto Colapicchioni e Luca Galuzzi

Dal registratore al drive del C64

Tutti i segreti delle memorie di massa del Commodore 64

Lire 7.000

Autori Vari

ADA

Il linguaggio passepartout dei computer degli anni '80.

Lire 5.000

Clizio Merli

Il linguaggio PASCAL

Un manuale tascabile per lo studio e la programmazione.

Lire 5.000

Sì, voglio arricchire la mia biblioteca con i seguenti volumi al prezzo di copertina + lire 3.000 per spese di spedizione.

- ☐ **64 Programmi per Commodore 64**
- ☐ **Strategie vincenti per i tuoi videogames**
- ☐ **62 Programmi per Vic 20 C16 e Plus77**
- ☐ **Utilities e giochi didattici**
- ☐ **Tutti i segreti dello Spectrum**
- ☐ **Simulazioni e test per la didattica**
- ☐ **Imparare giocando il basic dello Spectrum**
- ☐ **I miei amici C16 e Plus4**
- ☐ **Pascal per Commodore 128**
- ☐ **Dal registrattore al drive del C64**
- ☐ **ADA**
- ☐ **Il linguaggio Pascal**

Nome ....

via .... **N.ro.** .... **telefono** ....

CAP .... **Città** ....

Su tale importo mi praticherete lo sconto del 10% in quanto abbonato a ☐ *Commodore Computer Club* ☐ *Personal Computer* ☐ *Com puter* ☐ *VR Videoregistrare*. Pertanto vi invio la somma soltanto di lire ....

Valore dell'ordine lire..........

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Un ripasso sui cicli

Una buona conoscenza di come il proprio computer gestisce i cicli FOR...NEXT è parte essenziale del bagaglio tecnico di ogni buon programmatore; eppure i manuali sono spesso troppo stringati su tale punto; ad esempio si trova chiaramente detto che un programma (concettualmente assurdo) del tipo:

```
FOR X ...:
FOR Y ...:
NEXT X:
NEXT Y
```

non può funzionare; ma non è spiegato come fa il computer ad accorgersi dell'errore. La cosa è in realtà banale, pur di essere a conoscenza della:

REGOLA 1.
Un comando NEXT chiude automaticamente tutti i cicli eventualmente aperti dopo il corrispondente FOR.

Incontreremo via via in questo articolo altre informazioni analoghe.

Per controllare la vostra conoscenza sui cicli potete provare a svolgere i seguenti problemi:

(a) scrivere un pezzo di programma che realizzi un ciclo infinito senza fare uso di DO....LOOP, se il vostro BASIC ne è provvisto.
(b) scrivere un pezzo di programma che simuli un: DO...LOOP ⟨UNTIL condizione⟩; ovviamente, anche qui, senza usare i comandi DO, LOOP, UNTIL.
(c) scrivere un programma che scriva tutti i terni del gioco del lotto (una volta sola; cioè, se ha già scritto ad esempio 1,2,3 non deve poi scrivere 2,1,3.
(d) "parametrizzare" il programma precedente: il programma chiede un numero K tra 2 e 5; poi se K=2 scrive tutti gli ambi, se K=3 tutti i terni, se K=4 tutte le quaterne, se K=5 tutte le cinquine e questo è un po' più difficile dei precedenti!

Tutti i problemi proposti hanno varie soluzioni. In particolare ovunque si potrebbero evitare i cicli e risolvere il problema infarcendo il programma stesso di IF...THEN.

Tuttavia questo tipo di soluzioni è in generale poco efficiente e, se possibile, da evitare.

Ad esempio una "buona" soluzione al problema (a) è la seguente:

```
100 FOR X=1 TO 9E9
...
...
200 NEXT X
```

Se tra la riga 100 e la 200 non si sono inserite modifiche al valore di X (o dei GOTO che facciano saltare la linea 200) si è ottenuto un ciclo veramente infinito: quando infatti il valore di X arriva a circa 8.7E9 i successivi NEXT non ne modificano più il valore a causa di arrotondamenti e approssimazioni, ed il valore 9E9 non verrà mai raggiunto (elaborare per credere).

Se al programma precedente si aggiunge una linea del tipo:

```
199 IF ⟨condizione⟩ THEN X=10E9
```

si è risolto anche il problema (b). Ai più curiosi segnalo che il problema (a) poteva anche essere risolto sostituendo la linea 100 con la seguente:

```
100 FOR X=1 TO 2 STEP 0
```

Tuttavia questa variante si presta male ad una eventuale successiva compilazione: certi compilatori rifiutano, o compilano male, lo step 0 (compilare per credere), e rischia di dare risultati strani su computer non Commodore, che gestiscono in maniera diversa lo step 0. Per quanto riguarda il problema (b) la 199 potrebbe essere sostituita da:

```
199 IF ⟨condizione⟩ THEN FOR X=1 TO 1
```

in effetti, benchè i manuali non si dilunghino sull'argomento, si ha:

REGOLA 2.
Riaprendo un ciclo già aperto si ottiene automaticamente anche la chiusura del ciclo precedente: questa regola ha un'importante eccezione che vedremo tra poco.

Il problema (c) non dovrebbe aver fatto troppe vittime dal momento che l'unica difficoltà è la ricerca dei buoni estremi per i cicli. Una "buona" soluzione potrebbe essere:

```
10 FOR X1=1 TO 88
20 FOR X2=X1+1 TO 89
30 FOR X3=X2+1 TO 90
40 PRINT X1,X2,X3
50 NEXT X3,X2,X1
```

o anche:

```
50 NEXT:NEXT:NEXT
```

che è più veloce, anche se meno chiaro.

Per il problema (d) la via più ovvia è quella di scrivere quattro programmi distinti che realizzino rispettivamente la stampa degli ambi, dei terni, delle quaterne e delle cinquine, premettendo al tutto un comodo:

```
ON K-1 GOTO...
```

Un'altra via consiste nel modificare il programma relativo ai terni scrivendo cinque FOR e cinque NEXT, insieme ad un buon numero di IF che, in funzione del valore di K, facciano eseguire solo i cicli voluti, e printare solo i numeri corrispondenti.

Se avete scelto una di queste due vie significa che o siete vittime della incalzante propaganda a favore di linguaggi più strutturati del BASIC, oppure non conoscete l'eccezione alla regola 2 cui si è accennato poco fa: date un'occhiata al riquadro 1 e vi convincerete che il problema ammetteva soluzione banale, pur di sfruttare bene la (sia pur scarsa) ricorsività permessa dall'interprete Basic. La spiegazione è semplice se si studia la regola n.3

```
100 REM TUTTE LE COMBINAZIONI D
    EL GIOCO DEL LOTTO
110 N=90:INPUT K:H=1:GOSUB 200:
    END
120 :
200 IF H>K THEN FOR Y=1 TO K:PR
    INTX(Y);:NEXT:PRINT:RETURN
210 FOR X=X(H-1)+1 TO N-K+H
220 X(H)=X:H=H+1:GOSUB 200:H=H-
    1:X=X(H):NEXT:RETURN
```

Ecco un esempio di come si possano sfruttare le possibilità ricorsive del BASIC: il programma chiede un numero K tra 2 e 5, ed a seconda del valore di K visualizza tutti i possibili ambi (K=2) ovvero terni (K=3), quaterne (K=4) o cinquine (K=5) nel gioco del Lotto.

Naturalmente, volendo limitarsi a verificare il corretto funzionamento del programma, è bene diminuire di molto il valore di N in linea 100, altrimenti l'elaborazione per valori di K superiori a 2 richiede tempi lunghissimi!

Cambiando il valore di N in linea 100 si può adattare il programma per farne ad esempio il nucleo centrale di un programma che giochi a MASTERMIND, con un numero di "buchi" (dato da K) che non è fissato a priori: la linea 110 si semplifica notevolmente se si vogliono ammettere le ripetizioni di colori, e si complica un po' in caso contrario.

Su C/128 un programma di questo tipo gira fino a K=22, mentre su Vic-20 e C/64 il programma va "OUT OF MEMORY" già per K=6; tuttavia la versione compilata (con AUSTRO-SPEED) di questo tipo di programmi gira fino a K=10. Il compilatore, probabilmente, non memorizza in stack il numero della linea che chiama la subroutine, risparmiando così due byte per ogni chiamata.

REGOLA 3.
Un ciclo FOR aperto in una subroutine chiude eventuali cicli con lo stesso nome già aperti nella stessa subroutine; ma non tocca i cicli in sospeso del programma principale nè quelli aperti in subroutine "più esterne".

Ricordiamo infine un'ultima regola:

REGOLA 4.
Il comando RETURN chiude automaticamente TUTTI i cicli aperti nel corso della subroutine corrispondente!

Cenni sulla struttura LIFO

Tra gli strumenti concettualmente più importanti che ogni computer deve gestire, figura lo STACK (in italiano: CATASTA), cioè una struttura L.I.F.O.: Last In First Out, ovverosia il primo ad uscire è quello che era arrivato per ultimo.

Nello svolgimento dei suoi compiti l'interprete ha spesso bisogno di un "quaderno di brutta", in cui annotare e memorizzare informazioni che saranno utilizzate in seguito. Una tipica situazione in cui si presenta questa necessità è la valutazione di un'espressione. Ad esempio per eseguire:

```
PRINT 2+3*4-5
```

il computer dovrà limitarsi a memorizzare i dati incontrati fino a che non arriva al segno meno (-). Se, dopo il 4, fosse scritto il simbolo di elevazione a potenza, non si potrebbe ancora operare...

Arrivato al segno meno, può finalmente calcolare qualcosa: gli ULTIMI dati memorizzati (il numero 3, la moltiplicazione ed il numero 4) possono essere sostituiti con il valore 12, risultato dell'operazione parzialmente eseguita; poi si riprende il lavoro.

Da questa sommaria descrizione dovrebbe essere chiaro il perchè della struttura LIFO: di volta in volta si deve lavorare con gli ultimi dati memorizzati!

Fisicamente lo stack dei computer Commodore, almeno nei "vecchi" modelli (Vic-20 e C/64) è realizzato "in pagina 1": alcune delle memorie da 256 a 511 vengono appunto utilizzate per memorizzare le informazioni in sospeso: la prima informazione è scritta (pokata) in locazioni alte, le successive in celle di indirizzo via via più basso (tipicamente: si parte dalla cella 504, poi si lavora sulla 503, poi sulla 502 e così via).

Uno dei registri interni del microprocessore (detto usualmente registro S)"punta" costantemente alla cella "in cima allo stack". Quando si scrive qualcosa, "S" viene decrementato, mentre quando si legge qualcosa "S" viene incrementato, in modo da puntare sempre alla prima informazione da elaborare, che è appunto l'ultima di quelle immesse e non ancora utilizzate.

Si faccia attenzione al fatto che lo stack è "a testa in giù": come si è già detto l'operazione di scrittura abbassa il valore di S e quella di lettura lo innalza. Si osservi anche che, al contrario di quanto faremmo sul "quaderno di brutta", le informazioni utilizzate non vengono cancellate: il solo fatto che il registro S punti più in alto è sufficiente a non tenerne più conto.

I comandi GOSUB e RETURN

Tra le strutture per le quali uno stack si rivela indispensabile c'è quella di subroutine. Quando l'interprete incontra il comando GOSUB vengono sviluppate nell'ordine le seguenti fasi (si veda la figura 1):

(1) si guarda se nello stack c'è posto sufficiente per le operazioni che seguiranno; in caso contrario viene emesso un messaggio ("OUT OF MEMORY ERROR") che forse può essere causa di confusione: non è finita la zona di memoria riservata al programma ed alle variabili, ma quella dedicata allo stack!

(2) viene ricopiato nello stack il valore contenuto nella cella 123, poi quello contenuto nella cella 122 (gestite automaticamente dall'interprete); si tratta di parte alta e parte bassa del punto in cui si sta lavorando al momento del "riconoscimento" del GOSUB (cioè la cella successiva a quella che contiene GOSUB);

(3) viene immesso nello stack il contenuto della cella 58, poi quello della cella 57 (cioè parte alta e bassa del numero di linea in elaborazione);

(4) viene immesso nello stack il numero 141, token del comando GOSUB;

(5) si chiama la subroutine che gestisce il comando GOTO; essa provvede a leggere il numero che segue il GOSUB, a cercare l'indirizzo della linea in questione e ad immettere tale indirizzo nelle celle 122 e 123 che vengono gestite poi dall'interprete;

(6) si salta alla routine dell'esecutore, che provvederà a riprendere l'esecuzione dall'indirizzo costruito nella fase (5).

Il lettore attento, se non è esperto di **LM** (Linguaggio Macchina), dovrebbe a questo punto protestare: in fase (5) si è parlato di subroutine...; ed in effetti anche il **LM** possiede una struttura analoga a quella del GOSUB- RETURN del BASIC: come mnemonico per tali comandi sono usati i termini JSR e RTS. Torneremo più oltre su tale punto: per ora limitiamoci a tenere presente che anche la routine dell'esecutore, chiamata in fase (6), fa uso dello stack, depositandovi due numeri; si veda ancora la figura 1.
Inversamente, quando si incontra il comando RETURN:

(1) si controlla che, tolti due numeri dallo stack, in cima allo stack si trovi il valore 141, sennò si emette il messaggio "RETURN WITHOUT GOSUB"; ma c'è un'eccezione che vedremo tra poco.

(2) si prelevano i 2 valori successivi dallo stack, immettendoli rispettivamente nelle memorie 57 e 58;

(3) i due valori ancora successivi vengono ricopiati nelle memorie 122 e 123;

(4) ora, ridando la mano all'esecutore, questi lavorerebbe a partire dalla cella successiva a quella che conteneva il GOSUB, provocando errore...; perciò, invece di passare direttamente la mano all'esecutore, si salta alla routine del DATA: tale routine provvede ad incrementare il puntatore 122 e 123 fintantochè non incontra un 58 (token dei due punti) oppure uno 0 (token di fine linea). Ora l'esecuzione può riprendere senza problemi.
A questo punto il lettore non dovrebbe avere difficoltà a capire perchè frasi del tipo:

GOSUB 100 E AL RITORNO NON DARMI UN SYNTAX ERROR

vengono elaborate senza errori; così come dovrebbe rendersi facilmente conto del perchè, nella memorizzazione degli indirizzi di ritorno dalle subroutine, la struttura LIFO sia indispensabile per una corretta gestione di programmi del tipo:

```
100 GOSUB 200
110 ...
.
199 END
200 .....
210 GOSUB300
220 ...
.
290 RETURN
300 ...
.
400 RETURN
```

Si tratta di "nested subroutine": se non si usasse una struttura LIFO per gli indirizzi di ritorno, la linea 400 rinvierebbe l'esecuzione alla riga 110 anzichè alla 220!

Naturalmente ogni chiamata di subroutine "gonfia" lo stack, inserendo in esso nuove informazioni. Questo è il motivo per cui il programma:

```
10 N=N+1:PRINT N;:GOSUB10
```

si ferma ben presto col messaggio "OUT OF MEMORY"; sul C/64 e sul Vic-20 si arriva a scrivere il valore 24; nel riquadro 2 sono fornite altre informazioni relative alla "capienza massima".

```
100 PRINT"FARE RUN 200, RUN 300
     E RUN 400"
110 PRINT"SUI NUOVI MODELLI FAR
    E ANCHE RUN 500 E RUN 600":
    END
199 :
200 N=N+1:PRINTN;:GOSUB 200
210 REM VIC/20 E C/64 REGGONO F
    INO A N=24
220 REM C/16 E PLUS/4 FINO A N=
    40
230 REM C/128 FINO A N=67,POI D
    A' "FORMULA TOO COMPLEX" AN
    ZICHE' "OUT OF MEMORY"
299 :
300 N=N+1:FOR X=1 TO 2:PRINTN;:
    GOSUB 300
310 REM VIC/20 E C/64 REGGONO F
    INO A N=7
320 REM C/16 E PLUS/4 FINO A N=
    8
330 REM C/128 FINO A N=22
399 :
400 FOR X0=1 TO 2:FOR X1=1 TO 2
    :FOR X2=1 TO 2:FOR X3=1 TO
    2
410 FOR X4=1 TO 2:FOR X5=1 TO 2
    :FOR X6=1 TO 2:FOR X7=1 TO
    2
420 FOR X8=1 TO 2:FOR X9=1 TO 2
    :REM VIC/20, C/64, C/16 E P
```

```
    LUS/4  REGGONO FINO QUI
430 FOR XA=1 TO 2:REM E QUI VAN
    NO OUT
440 FOR XB=1 TO 2:FOR XC=1 TO 2
    :FOR XD=1 TO 2:FOR XE=1 TO
    2
450 FOR XF=1 TO 2:FOR XG=1 TO 2
    :FOR XH=1 TO 2:FOR XI=1 TO
    2
460 FOR XJ=1 TO 2:FOR XK=1 TO 2
    :FOR XL=1 TO 2:FOR XM=1 TO
    2
470 FOR XN=1 TO 2:FOR XO=1 TO 2
    :FOR XP=1 TO 2:FOR XQ=1 TO
    2
480 FOR XR=1 TO 2:REM IL C128 R
    EGGE FINO QUI
490 FOR XS=1 TO 2:REM E QUI VA
    OUT ANCHE LUI
499 :
500 DO:N=N+1:PRINTN;:FOR X=1 TO
     2:GOSUB 500
510 REM C/16 E PLUS/4 REGGONO F
    INO A N=7
520 REM IL C/128 REGGE FINO A N
    =19
599 :
600 DO:N=N+1:PRINTN;:GOTO 600
610 REM C/16 E PLUS/4 REGGONO F
    INO A N=39
620 REM IL C/128 REGGE FINO A N
    =102
```

Oltre a subroutine e cicli FOR-NEXT, lo stack dei nuovi modelli Commodore deve gestire anche i cicli DO-LOOP; probabilmente per tale motivo per lo stack non si usa più la sola pagina 1: sul C/128 si usano anche le pagine 8 e 9, mentre su C/16 e PLUS/4 parte di pagina 6 e parte di pagina 7. In particolare il livello di nidificazione ottenibile per FOR e GOSUB è più elevato. Il listato indica alcuni test significativi in proposito, fornendo nelle REM i valori in corrispondenza ai quali si verifica un "OUT OF MEMORY".

Una precisazione importante va fatta a proposito della fase (1) della gestione del RETURN: se da quando è stata chiamata la subroutine a quando si incontra il RETURN si fosse immesso qualche altra cosa nello stack, come si fa a "risalire" alla giusta posizione in cui dovrebbe trovarsi il token 141?

La risposta è semplice: nello stack possono essere state inserite molte cose (ad esempio risultati di conti intermedi per valutare un'espressione), ma al momento di eseguire il RETURN tutte queste informazioni sono state rilette, quindi il registro S "deve" puntare al 141... con un'unica eccezione: il caso in cui, durante l'esecuzione della subroutine, siano stati aperti, e non ancora chiusi, dei cicli.

Come si vedrà tra poco, ogni ciclo FOR occupa 18 posti nello stack. Se un ciclo non è stato chiuso, l'ultimo dei dati immessi in stack è il numero 129 (codice del FOR, così come il 141 codifica il GOSUB). Nella fase (1) del RETURN basta perciò cercare il 141 in cima allo stack; oppure 18 posti più in alto pur di aver incontrato un 129; o ancora 18 posti più in alto se si è incontrato un altro 129; e così via.

In particolare, se l'operazione va in porto, è chiaro che tutti i cicli aperti nella subroutine ed ancora in sospeso vengono automaticamente soppressi dal RETURN (si riveda a tal proposito la regola N.4).

La gestione dei cicli

Anche i cicli FOR...NEXT sono gestiti tramite stack; ogni ciclo necessita anzi di maggiore spazio che non una subroutine, essendo necessario memorizzare, oltre all'indirizzo di inizio del ciclo ed al numero di linea relativo, anche la variabile di ciclo, il valore dello step, ed il valore finale del ciclo.

Il tutto è condensato in 17 celle, seguite poi da una 18-esima in cui viene inserito il numero 129, token di FOR, che serve a segnalare l'esistenza di un ciclo; si dia un'occhiata alla figura 2. Naturalmente con blocchi da 18 celle ciascuno si fa presto ad esaurire la poca memoria riservata allo stack; si veda ancora il riquadro 2.

Per quanto riguarda l'esecuzione di un'istruzione FOR le fasi sono ormai semplici da descrivere:

(1) si legge il nome della variabile e si esegue un LET per attribuirle il valore di inizio ciclo;

(2) si legge l'elemento in stack puntato da S; se non è 129 si passa alla fase (4)

(3) si controlla, risalendo ogni volta di 18 posizioni, e solo se si continua ad incontrare il numero 129, se in stack è già presente un ciclo con tale nome; in caso affermativo si cambia il valore di S in modo che punti subito al di sopra di tale ciclo (annullando in tal modo il ciclo stesso e tutti quelli intermedi); in caso contrario si lascia S inalterato.

(4) si controlla che in stack ci sia posto sufficiente per un altro blocco da 18 celle; in caso contrario si emette il messaggio "OUT OF MEMORY".

(5) Salvo le ovvie eccezioni (se dopo il valore iniziale per il ciclo non si trova un "TO" si emette un SYNTAX ERROR; se non appare uno STEP si assume 1 come valore di step e così via) 18 caselle nello stack vengono riempite come descritto in figura 2.

In particolare, da quanto detto nella fase (2), ci si può rendere conto di come, durante una subroutine, non si ha accesso ai cicli aperti al di fuori della subroutine stessa (re-

gola 3); il che rende appunto possibili trucchi "ricorsivi" come quello descritto nel programma del riquadro 1. Analogamente, da quanto detto nella fase (3), si può controllare la validità della regola 2.

Anche la descrizione di quanto avviene quando si esegue un NEXT è ormai facile; precisamente:

(1) si vede se in stack c'è il valore 129 (in caso contrario si emette il messaggio "NEXT WITHOUT FOR").

(2) se il NEXT non è seguito da un nome di variabile, si opera come descritto nella seguente fase (4) relativamente al ciclo in cima allo stack.

(3) se la variabile che segue il comando NEXT non è quella del ciclo in cima allo stack, si risale di 18 posti (annullando in particolare il ciclo "saltato"; confronta regola 1) e così via fino a che si trova il ciclo giusto (e allora si va alla fase (4), col valore di S incrementato) o si trova un valore diverso da 129 (e allora si va alla routine di errore col messaggio "NEXT WHITHOUT FOR").

(4) si incrementa la variabile di ciclo tramite la quantità step (letta dallo stack); in funzione del valore estremo del ciclo, presente anch'esso in stack, si decide poi se l'esecuzione deve essere trasferita all'inizio del loop, di cui si è memorizzato l'indirizzo ed il numero di linea, proprio per poter eseguire tale salto; qui il salto si esegue semplicemente trasferendo l'indirizzo voluto nelle celle 122 e 123.

(5) se invece il ciclo deve essere chiuso ci si limita a incrementare di 18 il valore di S, senza modificare l'indirizzo in 122 e 123; però, prima di passare la mano all'esecutore, si controlla se il testo da elaborare continua con una virgola (per trattare sintassi del tipo NEXT X,Y).

Che cos'altro bolle in pentola?

Naturalmente anche i programmi interni del computer fanno un uso intensivo dello stack. In generale perciò nello stack si troveranno varie informazioni, da utilizzare nell'ordine corretto. Esemplifichiamo la cosa al livello JSR-RTS che, come si è detto, è l'analogo in LM del GOSUB-RETURN.

Il microprocessore possiede una coppia di registri interni cui si dà il nome di PC (Program Counter, cioè contatore di programma); essa fornisce, istante per istante, l'indirizzo della cella da esaminare, svolgendo così un ruolo analogo a quello svolto, per l'interpretazione di un programma BASIC, dalla coppia di celle 122, 123 (si riveda la fase (2) del GOSUB).

Invertendo i termini del problema, cominciamo a vedere cosa accade quando si incontra l'analogo del RETURN (indicato col simbolo RTS, e codificato col numero decimale 96, esadecimale 60): quando la cella di memoria puntata da PC contiene il numero 96 il microprocessore si limita a prelevare dallo stack i primi due valori, immettendoli nel registro PC; l'effetto di un RTS è perciò quello di spostare l'elaborazione all'indirizzo memorizzato in cima allo stack.

Se si confronta quest'operazione con quella descritta in occasione di un RETURN ci si accorge che si tratta dello stesso tipo di operazione, con due sole differenze: da un lato il LM non possiede "numeri di linea"; e dall'altro è eliminato ogni controllo (del tipo: c'era un 141 sullo stack?); è quindi compito del programmatore far si che, quando verrà incontrato un RTS, in cima allo stack si trovi effettivamente l'indirizzo cui si vuole saltare...

Questa assenza di controlli costituisce d'altronde croce e delizia del LM: croce perchè obbliga il programmatore ad un'estrema attenzione; delizia perchè, eliminando ogni controllo, l'esecuzione procede a velocità notevolmente maggiore!

In maniera analoga, quando la cella in esecuzione contiene il numero decimale 32 (codice dell'istruzione JSR, esadecimale 20) il lavoro svolto consiste nello scrivere sullo stack i valori di PC (è l'indirizzo cui si dovrà tornare!); e nell'immettere in PC l'indirizzo che si trova dopo il JSR (naturalmente il JSR è seguito da un indirizzo, e non da un numero di linea); confrontando con quello che si è costretti a fare in occasione di un GOSUB ci si rende conto che, ancora una volta, l'idea base è la stessa, ma la realizzazione è sfrondata di tante "parti accessorie" che, sia pur utili in BASIC per un'efficiente diagnostica, non hanno motivo di esistere al livello del LM.

In realtà niente obbliga ad usare un RTS solo in coppia con un precedente JSR: anzi, molto spesso quello che in BASIC si scriverebbe:

GOSUB XXX: GOTO YYY

viene simulato in LM con la seguente tecnica: si immette nello stack l'indirizzo cui si vuole saltare al termine della subroutine; poi si esegue non un JSR, ma semplicemente un JMP (equivalente del GOTO) all'indirizzo della subroutine; l'RTS finale della subroutine rinvierà l'esecuzione all'indirizzo preparato nello stack...

Si tratta di un trucco che è usato molto spesso nel BASIC Commodore; ad esempio per gestire il comando SYS, un altro dei comandi Basic dei quali i manuali dicono veramente troppo poco. Salvo il caso del C/128, dove la gestione è un po' diversa, il comando SYS forza il Computer ad eseguire le operazioni:

(i) valutazione dell'espressione che segue il comando SYS (che deve essere compresa tra 0 e 65535); la parte intera di tale valore viene immessa nelle memorie 20, 21;

(ii) immissione nello stack di un "indirizzo di ritorno", variabile a seconda del modello; ad esempio sul C/64 l'indirizzo in questione è $E146;

(iii) immissione in quattro registri interni (indicati usualmente con A, X, Y, P) dei valori contenuti rispettivamente nelle celle 780, 781, 782, 783 (che chiameremo "registri di passaggio dati"; su C16 e PLUS/4 i registri di passaggio dati sono le memorie 1365, 1366, 1367, 1364);

(iv) esecuzione di un JMP all'indirizzo contenuto nelle memorie 20, 21 già viste in (i).

Naturalmente il programmatore, prima dell'esecuzione del comando SYS, deve aver provveduto a scrivere, a partire dall'indirizzo della SYS, il suo programma in LM; e, se necessario, a preparare i valori che gli servono nei registri di passaggio dati. La scrittura del programma in LM non sarà necessaria se si fa una SYS a un indirizzo ROM, ma è comunque indispensabile che la routine chiamata tramite SYS termini con un RTS, cosicchè, quando la routine è terminata, l'RTS possa mettere in azione il procedimento preparato nella fase (ii).

L'esecuzione verrà allora trasferita all'indirizzo preparato in cima allo stack, che corrisponde alla routine che gestisce la fase finale della SYS: questa esegue le operazioni inverse di quelle indicate in (iii), cioè il contenuto dei registri interni A, X, Y, P è "pokato" nei registri di passaggio dati (da cui, tramite PEEK, l'utente potrà leggerli).

Naturalmente, se l'utente non ha bisogno di leggere i valori finali dei registri interni, può eliminare queste fasi terminali della SYS semplicemente inserendo nel suo programma in LM due comandi PLA (codice 104; significa togliere dallo stack un valore, incrementando automaticamente il registro S); in generale tale operazione è utile per non gonfiare inutilmente lo stack; mentre diventa obbligatoria quando si voglia "riorganizzare" lo stack...

Ad esempio supponiamo di voler eseguire un "CLEAR STACK", cioè di voler annullare tutti i cicli e le subroutine in sospeso; è una delle (tante) operazioni che esegue il comando CLR del BASIC, quindi si può sperare di limitarsi a fare una SYS in ROM, alla subroutine che appunto riporta il registro interno S a puntare alla base dello stack.

La subroutine in questione è all'indirizzo $A67A (decimale 42618) sul C/64; al solito, sul Vic-20, è $2000 più avanti; però, facendo direttamente SYS 42618, la presenza in stack dell'indirizzo di ritorno della SYS genera un malfunzionamento; occorre scrivere da qualche parte in RAM un (mini)programma in LM:

PLA
PLA
JMP$A67A
(ovvero JMP$C67A sul VIC)

e con una SYS a tale programma si ha l'effetto voluto.

Nel riquadro 3 è risolto (essenzialmente in BASIC) un problema più difficile: la subroutine fornita permette di costruire il comando POP, eliminando cioè dallo stack il blocco relativo all'ultima subroutine aperta.

Concludiamo osservando che la struttura della SYS non prevede la possibilità di lavorare col registro interno S; sebbene la scrittura in tale registro sia un'operazione pericolosa, che rischia di "inchiodare" il computer, la lettura del contenuto di S potrebbe essere interessante (il programma del riquadro 3 ne ha già fatto uso).

Il riquadro 4 fornisce un trucco banale per eseguire tale lettura.

Nel riquadro 5 è fornito un esempio di utilizzo delle informazioni ora viste relative al comando SYS, con una routine che, sfruttando le informazioni contenute nelle celle 61, 62, risolve il problema del GOTO CALCOLATO in maniera leggermente diversa da quella proposta su CCC N.31, rubrica "Nuovo Sistema". Ancora sull'uso delle locazioni 61, 62 è basata la utility del riquadro 6: un "Delete da un certo punto in poi".

```
40000 POKE  74,255:SYS 41866:REM
        SU VIC20 FARE SYS 50058
40010 RS=PEEK(781)+264:IC=PEEK(RS
      -4)+256*PEEK(RS-3)
40020 F=PEEK(RS)=141:IF F=0 THEN
      RS=RS-7
40030 FOR IC=IC TO 9E9:IF PEEK(IC
      )<>58 THEN NEXT
40040 IC=IC+1:POKE RS+4,IC/256:PO
      KE RS+3,IC-PEEK(RS+4)*256:R
      ETURN
```

A volte può far comodo annullare l'ultima subroutine in sospeso (oltre, ovviamente, ai cicli in essa aperti) senza che l'elaborazione venga trasferita (come avverrebbe eseguendo un RETURN) al punto in cui si era fatto il corrispondente GOSUB.

Stiamo parlando del comando POP, sfortunatamente assente nel BASIC Commodore. La cosa è delicata, perchè si deve in particolare controllare che qualche subroutine sia effettivamente stata aperta; e in tal caso, oltre all'ultima subroutine aperta, occorre anche chiudere tutti i cicli eventualmente aperti in essa.

Una routine in LM che esegua tale operazione non è difficile, ma si può anche risolvere il problema operando essenzialmente in BASIC, come mostra il segmento di programma accluso (i numeri di linea ed i nomi delle variabili possono essere cambiati a piacere).

Dovunque nel resto del programma eseguendo:

GOSUB 40000: RETURN (MA POI TORNA QUI !)

l'esecuzione sarà comunque riportata ai puntini che seguono il RETURN; in più, se prima del GOSUB c'era qualche subroutine in sospeso, la variabile F vale -1 e l'ultima subroutine viene chiusa (insieme a tutti i cicli in essa aperti); in caso contrario F vale 0. Volendo eseguire un "POP multiplo" basta iterare l'operazione.

Esercizio per i più distratti: controllare che in questo riquadro è già stato detto più volte che una sintassi del tipo:

FOR X=1 TO 5:GOSUB 40000:RETURN: NEXT

darà risultati imprevedibili!

Iterando l'operazione fino a che non risulti F=0 si annullano tutte le subroutine in sospeso, ma non i cicli FOR eventualmente aperti nel programma principale; volendo annullare anche questi ultimi è più semplice fare uso del programma CLEAR STACK descritto nell'articolo.

N.B: Nel listato di questo riquadro, fate attenzione ad interpretare correttamente eventuali segnalazioni di errore.

Come detto nell'articolo il comando SYS permette di passare dei parametri dal BASIC al LM e viceversa; sul C/128 si tenga presente che i valori da passare possono essere direttamente "accodati" alla SYS, anzichè POK-ati; e le PEEK finali possono essere sostituite da un RREG.

Su C/16, PLUS/4 e C/128 c'è anche la possibilità di leggere il valore del registro interno S: dopo una SYS basta leggere PEEK(9) sul C/128 e PEEK(1368) sugli altri. Cosa si può fare su VIC-20 e C/64 per leggere il valore di S?

Sapendo che il LM possiede un comando che trasferisce il contenuto di S nel registro X (codice 185, mnemonico TSX) basterà ovviamente scrivere da qualche parte in RAM il mini-programma

TSX
RTS

e poi leggere il contenuto del registro 781; cioè pokare in due locazioni consecutive della RAM i numeri 185, 96 e, fatta una SYS a tale mini-programma, leggere PEEK(781).

Tuttavia non è necessario scrivere il sia pur breve programma in LM, se lo troviamo già scritto da qualche parte in ROM; ed in effetti sul C/64 basta fare SYS 57409 (gli esperti di LM provino a disassemblare la zona precedente, che è il trattamento di EXP; avranno una sorpresa...) mentre per il VIC/20 la coppia 185, 96 non si trova; c'è tuttavia in $FE53, decimale 65107, come parte della routine di gestione parametri per l'apertura di un file, la routine:

TSX
STY$B9
RTS

(ed anche qui, disassemblando "bene", si avrà una sorpresa) che può essere utilizzata da BASIC preparando per il registro Y un valore che renda innocuo lo STY$B9; basta eseguire:

POKE 782,PEEK(185):SYS65107

seguito poi dalla lettura della locazione 781.

```
10000 A=PEEK(61)+256*PEEK(62):B=P
      EEK(A+1)+256*PEEK(A+2)-10:R
      EM *********
10001 POKE 66,A/256:POKE 65,A-256
      *PEEK(66)
10002 FOR X=B TO B+8:READ C:POKE
      X,C:NEXT:END
10003 DATA 104,104,132,21,134,20,
      76,163,168

10004 POKE 782,N/256:POKE 781,N-2
      56*PEEK(782):SYSB
```

Questo segmento può essere inserito ovunque in un programma (i numeri di linea ed i nomi delle variabili possono essere cambiati a piacere); si esegua RUN 10000, poi si cancellino le linee 10001, 10002, 10003; provando a listare la linea 10000 si otterrà ora qualcosa di strano, ma non c'è da preoccuparsi...

A questo punto, dovunque nel resto del programma, il comando "GOTO 10000" equivarrà a "GOTO N"; mentre il comando "GOSUB 10000" equivarrà a "GOSUB N".

La linea 10000 fornisce nella variabile B l'indirizzo della prima cella dopo la REM; la linea 10001 esegue semplicemente un RESTORE 10000, così da leggere i dati della linea 10003, e non qualche altra cosa dal resto del programma; infine la 10002 sostituisce gli asterischi di linea 10000 con un breve programma in LM . Una volta svolto il loro compito le linee 10001, 10002, 10003 vengono eliminate.

Quando, nel resto del programma, si esegue un GOTO 10000 o un GOSUB 10000, la linea 10000 fornisce in B l'indirizzo per la SYS e la linea 10004 prepara come valori da trasmettere ai registri interni X ed Y la parte bassa e parte alta del numero N; dopodichè si fa una SYS all'indirizzo B calcolato in linea 10000.

La parte in LM corrisponde a:

PLA
PLA
STX$14
STY$15
JMP$A8A3

(al solito, su VIC-20, si aumenterà di $2000 l'indirizzo del JMP, facendo JMP$C8A3; si tratta cioè di sostituire l'ultimo dato 168 di linea 10002 con il valore 200). Qui, al contrario di quanto avveniva per il programma CLEAR STACK, i due PLA iniziali potrebbero essere eliminati per accorciare ancor più la routine.

Sui nuovi modelli la ricerca della posizione del primo asterisco (*) è più complicata (le memorie 61, 62 operano in modo diverso); su C/16 e PLUS/4 si tenga inoltre presente del diverso indirizzo dei registri di passaggio, come indicato nell'articolo; il JMP va fatto alla locazione $8D50.

Su C/128, come ricordato nel RIQUADRO 4, la nuova sintassi del comando SYS permette di "accodare" direttamente alla SYS i valori da passare; però occorre modificare il programma in LM nella forma:

PLA
PLA
STX$16
STY$17
JMP$59E2

(si faccia inoltre attenzione ad usare preliminarmente il comando BANK).

Il sistema usato è di carattere generale: una qualunque routine in LM abbastanza corta da entrare in una linea BASIC può essere inserita in modo analogo in un programma; naturalmente la routine deve essere rilocabile (aggiunta o soppressione di linee BASIC ne modificheranno la posizione!); si deve però evitare di usare questo sistema quando i dati del programma in LM contengono degli zeri: uno 0 inserito in una linea BASIC provoca sempre dei guai ...

```
0 GOTO 8:REM PER NON DISTURBA
  RE IL PROGRAMMA PRINCIPALE
1 PRINT "DELETE DA  NNN  IN P
  OI":INPUT  "NNN";A
2 IF (A<6) OR (A>63999) OR (A
  <INT(A)) THEN STOP
3 PRINTCHR$(147):PRINT:PRINTA
  ;
4 PRINT "P=PEEK(61)+256*PEEK(
  62)+3:POKE64,P/256:POKE63,P
  -256*PEEK(64)"
5 PRINT A+1 "POKEP-1,0:POKEP-
  2,0:POKE45,PEEK(63):POKE46,
  PEEK(64):CLR:END"
6 PRINT "RUN" A CHR$(19);:POK
  E 198,6
7 FOR X=631 TO 633:POKE X,13:
  NEXT:POKE X,147:END
8 PRINT"IL DELETE SI LANCIA C
  ON :RUN 1"
```

Il BASIC di VIC-20 e C/64 non possiede il comodissimo comando DELETE (per l'eliminazione di un blocco di linee di programma). Se ci si accontenta di un "delete da un certo punto in poi" il listato qui accluso risolve il problema.

Ad esempio le linee qui proposte potrebbero essere inserite nel programma caricatore del "Nuovo Sistema": volendo caricare più parti, dopo la scrittura dei dati di ogni parte ed il relativo RUN basterà cancellare (eseguendo RUN 1) le linee da 1000 in poi, senza dover caricare di nuovo il programma caricatore; naturalmente in questo caso si soppprimeranno le linee 1 e 2 e si scriverà 1000 al posto di A nelle linee 3, 6; e 1001 al posto di A+1 in linea 5.

A tutti i Commodore Computer Club

Molti circoli si sono aperti e molti sono usciti allo "scoperto" dopo il nostro invito ad aprire un Computer Club, apparso sul N. 21 di C.C.C.

Allo scopo di rendere un servizio migliore ai nostri lettori che intendano contattare uno di questi simpatici circoli culturali, i segretari dei circoli stessi sono pregati di compilare il seguente tagliando, o sua fotocopia, e di inviarlo in busta chiusa (affrancata secondo le vigenti tariffe postali) a:

Systems Editoriale
Servizio Notizie Computer Club
Viale Famagosta, 75
Milano

La completa compilazione dell'intera scheda, è INDISPENSABILE per la pubblicazione gratuita sulla nostra rivista.

Nome del club: .... Sede del club: .... Via ....
C.A.P. .... Città .... Prov.: .... Tel. Prefisso: ......N. ....
Presidente: .... Segretario: .... N. soci fondatori: ....
N. di soci finora iscritti: .... Data di fondazione: .... Giorni di apertura della sede: ....
Orario di apertura: .... Computers disponibili (specificare): ....
Periferiche disponibili (specificare): .... Programmi disponibili (N. approssimativo): ....
Videogiochi N.: .... Professionali N.: .... Biblioteca tecnica N. volumi: ....
N. abbonamenti a riviste italiane: .... N. abbonamenti a riviste straniere: .... Quota di iscrizione L. ....
(Specificare se annuale, mensile ecc.) ....
Attività previste: .... Bollettino periodico emesso: .... Attività già svolte: ....
Eventuali sponsor: .... Disponibilità alla sponsorizzazione (sì/no): ....

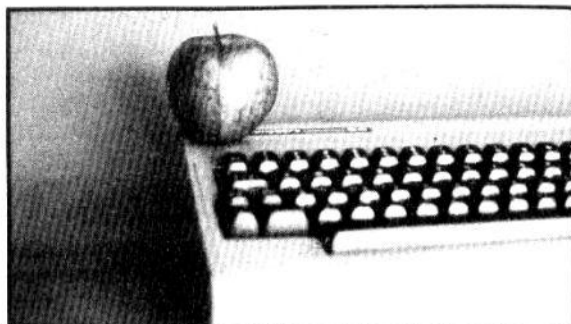



Figura 1
Lo stack del C/64 durante l'esecuzione di una subroutine.

Ogni comando GOSUB occupa sullo stack 7 byte; uno di essi contiene il numero 141, caratteristico del GOSUB; altri due forniscono il numero della linea BASIC che conteneva il GOSUB; altri due sono parte bassa e parte alta dell'"indirizzo di ritorno" (si tratta della prima cella successiva a quella in cui è tokenizzato il GOSUB).

I restanti due byte contengono anch'essi un indirizzo (in ROM) e puntano alla routine NSE (Next Statement Executor); sul VIC20 invece del 167 c'è un 199.

```
|-------------|   <-- cella 511
|-------------|
|-------------|
|-------------|
|-------------|
|-------------|
|-------------|
|=============|
|  Indirizzo  |   <-- parte alta
|--         --|
| di ritorno  |   <-- parte bassa
|=============|
|Numero (58)  |   <-- parte alta
|--  di     --|
|linea  (57)  |   <-- parte bassa
|=============|
|     141     |   <-- Token di
|=============|       GOSUB
|     167     |
|--         --|
|     233     |   <-- qui punta il
|=============|       registro S
|-------------|
|-------------|
|-------------|
|-------------|
|-------------|   <-- cella 256
```

```
|=============|
|  Indirizzo  |   <-- parte alta
|--         --|
| di ritorno  |   <-- parte bassa
|=============|
|   Numero    |   <-- parte alta
|--         --|
|  di linea   |   <-- parte bassa
|=============|
|             |
|--         --|
|  limite     |
|--         --|
| superiore   |   <-- 5 byte
|--         --|
| del ciclo   |
|--         --|
|             |
|=============|
| SGN di STEP |
|=============|
|             |
|--         --|
|   valore    |
|--         --|
|    dello    |   <-- 5 byte
|--         --|
|    step     |
|--         --|
|             |
|=============|
|  Indirizzo  |
|--         --|
|var.del ciclo|
|=============|
|     129     |   <-- qui punta il
|=============|       registro S
```

Figura 2
Lo stack durante l'esecuzione di un FOR.

Ogni ciclo FOR in sospeso occupa 18 byte nello stack; il valore 129 è caratteristico del FOR; il valore dello step e quello del limite superiore del ciclo sono memorizzati nella usuale forma floating point, che richiede appunto 5 byte; il byte "SGN di STEP" vale 1 se lo step è positivo; vale 0 se lo step è nullo; vale 255 se lo step è negativo.

Per un ciclo del tipo:

FOR X=A TO B STEP C

l'"indirizzo di ritorno" punta alla cella contenente i due punti che seguono la C; l'"indirizzo variabile del ciclo" punta al primo dei 5 byte in cui è memorizzata la variabile X.

Si osservi che, proprio perchè i valori di B e C vengono memorizzati nello stack, successive modifiche ai valori di tali variabili non avranno influenza sul ciclo.

Parlar di Sprite

Tre routine (più una) per meglio comprendere la gestione dei sotto-schermi del popolare C/64

A cura di Alessandro de Simone

16600 Accensione sprite
(Commodore 64)

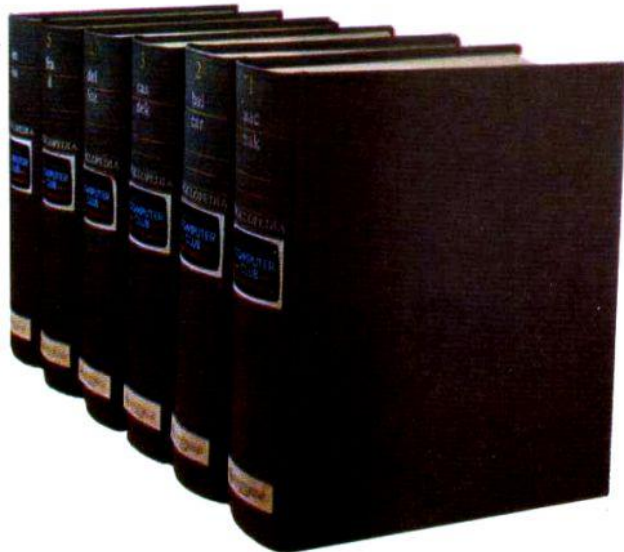
Questa routine gestisce l'accensione e il posizionamento di qualsiasi sprite (numerati da 0 a 7) in un qualsiasi punto dello schermo.

Nell'esempio d'uso vengono richiesti i parametri necessari per il corretto funzionamento della routine, che sono i seguenti (tra parentesi sono riportati i limiti minimo e massimo accettabili):

NS = numero sprite (0-7)
CS = colore sprite (0-255)
PX = posizione X (0-319)
PY = posizione Y (0-255)
EX = espansione X
EY = espansione Y

Per quanto riguarda le ultime due opzioni (EX, EY) va detto che si può rispondere con un qualsiasi numero (positivo o negativo) in quanto la routine non ne verifica il valore, ma solo la "diversità" dal numero zero. Ciò giustifica l'apparente incompletezza (If EY Then) nelle linee 16610 e 16612.

Ne approfittiamo, comunque, per ricordare che con alcuni compilatori la sintassi parziale viene accettata, ma anche il valore nullo è considerato "esistente" con il risultato che le istruzioni presenti dopo Then vengono sempre eseguite, qualunque sia il valore della variabile sottoposta alla verifica IF.

Per ciò che riguarda l'accensione si ricorre alla seguente formula:
POKE 53269,PEEK(53269)OR(2↑NS)
in modo da modificare il valore del registro accensione sprite senza spegnere gli altri sprite eventualmente accesi.

Infatti se per accendere lo sprite 2 avessimo eseguito POKE53269,2 avremmo acceso sicuramente lo sprite N.2, ma perso tutti gli altri eventualmente presenti.

Per quanto riguarda il posizionamento lungo l'asse X per valori superiori a 255, è bene sottolineare che è necessario settare il bit corrispondente allo sprite nel registro 53264 secondo le considerazione già fatte in merito all'accensione dello stesso sprite (funzione OR).

Nell'esempio d'uso si utilizza uno sprite a forma quadrata allocato, per comodità, nel buffer di cassetta (da 960 in poi, cfr. linea 200). Nulla vieta però, modificando i valori con una procedura Read-Data, di visualizzare uno sprite personalizzato dal lettore.

Per ulteriori chiarimenti sull'argomento, si consiglia vivamente la lettura dell'articolo "Tutto sugli sprite" (C.C.C. N.35).

```
  100 REM ESEMPIO D'USO
  110 REM PER SETTARE I PARAMETRI
       DI UN QUALSIASI SPRITE
  120 REM BY MICHELE MAGGI
  170 :
  200 FOR I=960 TO 960+63:POKE I,
      255:NEXT:PRINTCHR$(147);
  205 PRINTCHR$(19);
  210 INPUT "NUMERO SPRITE (0-7)"
      ;NS
  211 IF NS<0 OR NS>7 THEN 210
  220 INPUT "COLORE SPRITE (0-255
      )";CS
  221 IF NS<0 OR CS>255 THEN 220
  230 INPUT "POSIZIONE X";PX
  250 INPUT "POSIZIONE Y";PY
  260 INPUT "ESPANSIONE X";EX
  270 INPUT "ESPANSIONE Y";EY
  300 GOSUB 16600:GOTO 205
 9999 END
16600 IF PX>320 OR PX<0 OR PY<0 O
```

```
       R PY>255 OR NS<0 OR NS>7 TH
       EN X0$="ERR":RETURN
16601 POKE 53269,PEEK(53269) OR (
       2↑NS)
16602 POKE 53287+NS,CS:REM NS=N.S
       PRITE; CS=COLORE SPRITE
16603 IF PX>255 THEN P1=PX-255:PX
       =P1:POKE 53264,PEEK(53264)
       OR (2↑NS):GOTO 16605
16604 POKE 53264,PEEK(53264) AND
       (PEEK(53264)-(2↑NS))
16605 POKE 53248+(2*NS),PX:REM PX
       ,PY= COORDINATE SPRITE
16608 POKE 53249+(2*NS),PY
16610 IF EY THEN POKE 53271,PEEK(
       53271) OR (2↑NS):REM EY,EX=
       ESPANSIONE SPRITE
16612 IF EX THEN POKE 53277,PEEK(
       53277) OR (2↑NS)
16614 POKE 2040+NS,15
16618 RETURN
16620 :
16690 REM VARIABILI: NS,CS,PX,PY,
       EX,EY
16699 REM NOME: GESTIONE SPRITE
```

16700 Movimento sprite
(Commodore 64)

Questa routine ha lo scopo di visualizzare costantemente le posizioni X e Y di uno sprite controllato tramite i tasti cursore.

Anche per questa routine valgono le considerazioni fatte in precedenza in merito al registro 53264 (valore dell'ascissa maggiore di 255) che deve essere settato se lo sprite oltrepassa la posizione N.255.

Oltre ai valori X e Y viene anche visualizzato il comando POKE53264,FL dove FL assume valore 1 oppure 0 in funzione della posizione dello sprite.

```
100 REM ESEMPIO D'USO
110 REM MOVIMENTO SPRITE
120 REM CON TASTI CURSORE
125 :
130 PX=100:PY=100
140 FOR I=0 TO 62:POKE 832+I,25
     5:NEXT:POKE 2040,13
145 GET A$
150 IF A$="[DOWN]" THEN A=1:GOS
     UB 16700
151 IF A$="[UP]" THEN A=2:GOSUB
      16700
152 IF A$="[LEFT]" THEN A=3:GOS
     UB 16700
153 IF A$="[RIGHT]" THEN A=4:GO
     SUB 16700
154 IF A$=CHR$(13) THEN FOR I=1
      TO 10:PRINT"[DOWN]":NEXT:P
     RINTPX,PY,"POKE 53264,"FL:E
     ND
155 PRINT"[CLEAR]"PX,PY,"POKE 5
     3264,"FL:GOTO 145
210 :
9999 END
16700 V=53248:POKE V+21,1
16703 IF A<1 OR A>5 THEN X0$="ERR
       ":RETURN
16705 ON AGOTO 16710,16720,16730,
       16740
16710 PY=PY+1:IF PY>227 THEN PY=2
       27
16712 POKE V+1,PY:GOTO 16732
16720 PY=PY-1:IF PY<50 THEN PY=50
16722 POKE V+1,PY:GOTO 16732
16730 PX=PX-1:IF PX<25 AND FL=0 T
       HEN PX=25
16731 IF FL AND PX=0 THEN FL=0:PX
       =255:POKE V+16,0
16732 POKE V,PX:RETURN
16740 PX=PX+1:IF PX>254 THEN FL=1
       :PX=1
16742 IF FL THEN POKE V+16,1
16743 IF (PX>65) AND (FL>0) THEN
       PX=65:REM   RETURN
16745 POKE V,PX:RETURN
16748 REM MUOVE SPRITE CON TASTI
       CURSORE
16749 REM EVIDENZIANDONE LA POSIZ
       IONE
16799 REM ED IL VALORE DEL REGIST
       RO 53264
```

16800 Sprite scanner
(Commodore 64)

Utilizzando alcuni videogame (naturalmente sprotetti...) ci è capitato di osservare sprite molto belli. Interrompere un gioco sprotetto non è un problema (tasto di reset o altri sistemi): ma come fare per rintracciare gli sprite che hanno destato la nostra curiosità?

Può dunque capitare di voler individuare la posizione del blocco relativo agli sprite di vari programmi (sia in Basic che L.M.) sia per sapere in che porzione di memoria si allocano, sia per estrarre i 63 dati relativi a ciascuno di essi.

La routine proposta si incarica di risolvere il problema: sarà però dapprima necessario caricare il gioco che contiene gli sprite da esaminare; quindi resettare (non spegnere!) il computer per poi caricare e mandare in esecuzione il programma qui pubblicato.

Alla partenza verrà richiesto il numero del blocco che si intende esaminare; successivamente verranno visualizzati, su due file sovrapposte, tutti gli sprite, quattro in hi-res e quattro in multicolor, il cui disegno, ovviamente, varierà in funzione del valore del puntatore dello sprite.

I comandi possibili sono i seguenti:

Tasto più (+): incrementa il valore del blocco-sprite
Tasto meno (-): lo decrementa
Shift (+): incrementa il puntatore, ma di 64 unità alla volta
Shift (-): lo decremente di 64 unità alla volta
Clr/Home: azzera i puntatori e riparte.
X: espande gli sprite lungo X
Y: espande lungo Y
Freccia a sinistra: riporta gli sprite alle dimensioni originali.

Il motivo per cui vengono visualizzati gli otto sprite (che puntano, però, tutti allo stesso blocco indicato), è dovuto al fatto che, se ne avessimo utilizzato uno solo, questo rischiava di restare invisibile se avesse avuto lo stesso colore del fondo; analogamente rischiava di essere "incomprensibile" se visualizzato in multicolor, quando era da visualizzare in modo normale, e viceversa.

Con una routine in Basic, ovviamente, è possibile visualizzare i 256 sprite allocabili nel primo banco di memoria (da 0 a 16319): gli sprite che in vari programmi molto complessi (come Summer Games) vengono allocati sotto il Kernal o sotto il Basic non possono essere esaminati se non ricorrendo a notevoli modifiche che esulano dallo standard dell'Enciclopedia di routine.

```
  100 REM ESEMPIO D'USO
  110 REM VISUALIZZA SPRITE
  120 REM PRESENTI IN MEMORIA
  130 :
  150 INPUT "[CLEAR]PARTENZA";A:P
      RINT"PREMI UN TASTO"
  151 GET A$:IF A$="" THEN 151
  155 IF A<0 OR A>255 THEN 150
  280 PRINT"[CLEAR]BLOCCO"A; TAB(
      14)"LOCAZIONE"A*64
  290 PRINT"[17 DOWN]X - ESPANSIO
      NE IN X"
  300 PRINT"Y - ESPANSIONE IN Y"
  310 PRINT"← = RITORNO DIMENSION
      I NORMALI"
  320 PRINT"CLR = AZZERAMENTO BAN
      CO"
  340 GOSUB 16800
  350 GOTO 151
  360 :
 9999 END
16800 V=53248:POKE V+28,0:POKE V+
      21,255:POKE 33,0:POKE 650,1
      28:IF A$=CHR$(19) THEN A=0
16803 POKE V+28,240:POKE V,50:POK
      E V+1,70:POKE V+2,110:POKE
      V+3,70
16806 POKE V+4,170:POKE V+5,70:PO
      KE V+6,230:POKE V+7,70:POKE
       V+8,50:POKE V+9,120
16809 POKE V+10,110:POKE V+11,120
16810 POKE V+12,170:POKE V+13,120
16811 POKE V+14,230:POKE V+15,120
16814 IF A$="+" THEN A=A+1:IF A>2
      55 THEN A=255
16815 IF A$="-" THEN A=A-1:IF A<1
       THEN A=0
16820 IF A$="X" THEN POKE V+29,25
      5
16825 IF A$="Y" THEN POKE V+23,25
      5
16830 IF A$="←" THEN POKE V+23,0:
      POKE V+29,0
16835 IF A$="+" THEN A=A+8:IF A>2
      55 THEN A=255
16840 IF A$="|" THEN A=A-8:IF A<1
       THEN A=0
16842 POKE 198,0:FOR I=2040  TO
      2047:POKE I,A:NEXT:RETURN
16899 REM  NOME:SPRITE SCANNER
```

16900 Deek & Doke
(qualsiasi Commodore)

Nonostante sia già stata pubblicata una routine simile nella rubrica Enciclopedia L.M, ne proponiamo una in Basic, chiaramente più semplice.

Si tratta, in realtà, di due routine che, grazie alla brevità, sono unite in un corpo unico.

DEEK funge da doppia PEEK e serve per esaminare quale sia la locazione indicata da un qualsiasi puntatore.

In input viene richesto il primo puntatore; il valore di questo, e di quello che lo segue, viene quindi elaborato e visualizzato in output insieme con il valore della locazione puntata.

Ad esempio, se consideriamo il puntatore 43 e 44 (inizio del programma Basic) risponderemo all'input semplicemente con 43; verrà fornito in output il valore del-

Elenco delle routine pubblicate

(Fra parentesi è riportato il numero sono apparse)

63923 rem 16500 drum per c/64 (38)
63924 rem 16400 draw low/res (38)
63925 rem 16300 print v/cont (38)
63926 rem 16200 plot low-res (37)
63927 rem 16100 integrali (37)
63928 rem 16000 equaz. mista (37)
63929 rem 15900 equaz. terzo gr. (37)
63930 rem 15800 derivata di funz. (37)
63931 rem 15700 scritte rotanti (37)
63932 rem 15600 convers. coordin. (36)
63933 rem 15500 logar. base quals. (36)
63934 rem 15400 conversione basi (36)
63935 rem 15300 semplif. frazioni (36)
63936 rem 15200 divis. con N decim.(36)
63937 rem 50100 directory (35)
63938 rem 15100 lampeggio righe (35)
63939 rem 15000 frammenta schermo (35)
63940 rem 14900 delete window (35)
63941 rem 14800 cambia stringhe (34)
63942 rem 14700 slitta stringhe (34)
63943 rem 14600 ruota stringhe (34)
63944 rem 10500 input programmab. (34)
63945 rem 14500 scroll solo testo (33)
63946 rem 14400 sprite multiuso (33)
63947 rem 14300 zoom esadecimale (33)
63948 rem 14200 video orologio (33)
63949 rem 11100 funzioni inverse (32)
63950 rem 13200 centra messaggi (32)
63951 rem 14100 finestre di testo (32)
63952 rem 14000 gestione nome disk (32)
63953 rem 13900 load/save pg.video (31)
63954 rem 13800 scritte in ebcm (31)
63955 rem 13700 bit image mps/803 (31)
63956 rem 13600 or esclusivo (31)
63957 rem 13500 comandi extra prg (31)
63958 rem 13400 linee low-res. (31)
63959 rem 13300 elabora stringhe (31)
63960 rem 13200 centratura frase (32)
63961 rem 13100 menu con joy (30)
63962 rem 13000 menu con cursore (30)
63963 rem 12900 frase lampeggiante (29)
63964 rem 12800 bordo technicolor (29)
63965 rem 12700 fill memoria ram (29)
63966 rem 12600 text copy mps 803 (29)
63967 rem 12500 colore pag.testo (29)
63968 rem 12400 print using (31)
63968 rem 12400 print using (29)
63969 rem 12300 m.c.d. E M.C.M. (29)
63970 rem 50500 visualizza file (28)
63971 rem 50400 read file relativi (28)
63972 rem 50300 write file relat. (28)
63973 rem 50200 crea file relativi (28)
63974 rem 50000 legge blocks free (28)
63975 rem 12200 numeri congrui (28)
63976 rem 12100 protezione S/W (28)
63977 rem 12000 koala (27)
63978 rem 11900 change pag. video (27)
63979 rem 11800 salva ram (27)

(Le routine di questo numero sono di Michele Maggi)

la cella di memoria 43, quello della cella 44 e la locazione da essi puntata: 1, 8 e 2049.

DOKE simula una doppia Poke in funzione dell'inserimento della locazione che si vuole puntata dal puntatore "P", anch'esso specificato nell'input.

Se, ad esempio, vogliamo che il puntatore 43 e 44 punti alla locazione 49152, sarà sufficiente specificare la locazione (49152) e il puntatore (43); il computer risponderà visualizzando (ma non eseguendo) le istruzioni necessarie alla DOKE, in modo da evitare probabili e pericolosi errori indesiderati che potrebbero inchiodare il computer.

Per rendere operativi i due comandi Poke sarà sufficiente digitarli ed eseguirli a parte.

```
  100 REM ESEMPIO D'USO
  110 REM ESEGUE LE DOPPIE
  120 REM PEEK E POKE
  140 :
  160 INPUT "SCEGLI:DEEK (1) DOKE
       (2)";JK
  170 IF  JK=1 THEN INPUT "PUNTAT
      ORE";P
  180 IF JK=2 THEN INPUT "LOCAZIO
      NE DA PUNTARE";AD:INPUT "PU
      NTATORE";P
  185 :
  190 GOSUB 16900
  191 :
 9999 END
16900 IF JK<1 OR JK>2 THEN X0$="E
      RR":RETURN
16901 ON JKGOTO 16905,16918
16905 IF P<1 OR P>65535 THEN PRIN
      T"ERR":END
16910 PRINT"PEEK"P"="PEEK(P)
16912 PRINT"PEEK"P+1"="PEEK(P+1)
16914 PRINT"LOCAZIONE PUNTATA"PEE
      K(P)+256*PEEK(P+1)
16915 RETURN
16918 IF P<1 OR P>65535 THEN PRIN
      T"ERR":END
16919 IF AD<1 OR AD>65535 THEN PR
      INT"ERR":END
16920 HI=INT(AD/256)
16925 LO=AD-HI*256
16940 PRINT"POKE"P","LO
16941 PRINT"POKE"P+1","HI
16942 RETURN
16943 :
16945 REM ESEGUE LA DOPPIA PEEK E
       LA DOPPIA POKE
16999 REM  NOME: DEEK & DOKE
```

Come realizzare l'enciclopedia e utilizzarla nei propri listati.

Ai lettori che hanno acquistato per la prima volta questo numero di Commodore Computer Club, illustriamo qui di seguito, in breve, i vantaggi derivanti dalla raccolta proposta. Questa, a pensarci bene, è la versione "superiore" della rubrica "1 RIGA" e potrebbe anche denominarsi... "Una schermata"!

Oltre che utili per costituire un'enciclopedia, i brevissimi sottoprogrammi pubblicati su ogni numero, sono anche validissimi strumenti di studio per coloro che desiderano approfondire le proprie conoscenze del Basic, esaminando, senza fatica, particolari routine o insolite tecniche di programmazione.

- Dato che può esser "chiamata" più di una volta nel corso di un programma, nessuna routine contiene istruzioni del tipo DATA oppure DIM, allo scopo di non creare confusione col listato principale.
- Nessuna routine può far riferimento ad altre routine dell'enciclopedia.
- Nessuna routine può contenere variabili "banali" (A, A$, eccetera), ma solo variabili poco usate (X1$, X8, Y0%, eccetera).
- Ogni routine deve apparire, **per intero,** sullo schermo del computer e consentire, proprio per questo motivo, di essere esaminata comodamente.
- Ogni routine deve esser numerata secondo uno standard che ha la particolarità di esser ricordato facilmente:

Righe	Contenuto
XXY00	Prima riga del sottoprogramma
XXY89	Ultima riga utile del sottoprogramma
XXY90 REM	Prima riga di spiegazioni
XXY99 REM	Nome della subroutine

in cui XX sono due valori variabili da 10 a 63; Y è un carattere numerico compreso tra 0 e 9.

Qualsiasi subroutine, in altre parole, inizia con un numero, di cinque caratteri, che termina **sempre** con "00". La stessa subroutine, d'altra parte, ha l'ultima riga numerata con "99". Digitando, ad esempio: LIST 10800-10899
si avrà la certezza di veder apparire sullo schermo, **per intero,** la routine il cui nome si trova nella riga 10899.

Prima di accedere alla routine, è necessario assegnare, alle variabili indicate con REM da riga XXY89 a XXY98, particolari valori per il suo corretto funzionamento. Al "ritorno" una o più variabili conterranno il risultato dell'elaborazione.

In questo modo, per esser più chiari, è possibile simulare alcuni comandi di versioni Basic avanzate oppure, addirittura, creare nuove e inedite istruzioni. Ad esempio, il comando: SOUND 1,800,500
che, nel C-16, riproduce un suono di tonalità 800 tramite la voce 1 per la durata 500, potrebbe venir riprodotta, in un'ipotetica subroutine per il Commodore 64, con: X1=1:X2=800:X3=500:GOSUB12400
nell'ipotesi, ovviamente, che la routine in oggetto sia allocata da riga 12400 a 12499.

I listati pubblicati "girano" su ogni computer, salvo dove indicato diversamente.

E' ovvio che nel caso del Vic-20, (che, come è noto, ha uno schermo di soli 506 caratteri), le subroutine "universali" funzionano correttamente, ma non possono apparire per intero in una sola schermata.

Per quanto riguarda la digitazione, si tenga presente che sulla rivista, per motivi di chiarezza, i comandi e le istruzioni Basic sono separati tra loro da spazi bianchi. Nel digitare le linee di programma, pertanto, è opportuno ignorarli altrimenti si rischia di non restare in una sola schermata. Se, per esempio, leggete:

```
12100 X1=34:  X2  = SQR(X3) + LOG(X1)
```

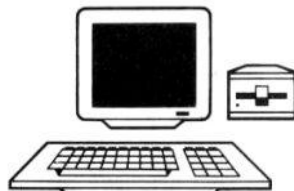
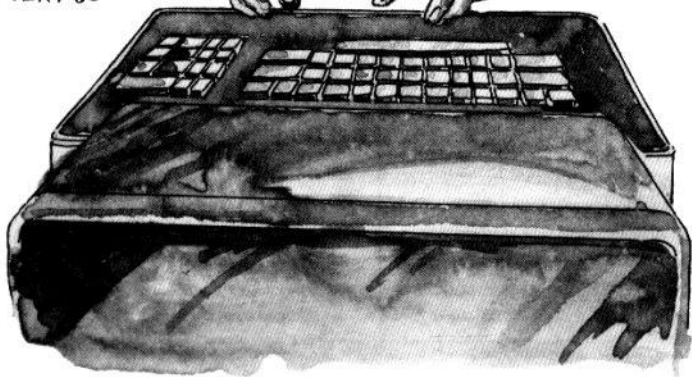
digitate nel modo seguente:

```
12100 X1=34:X2=SQR(X3)+LOG(X1)
```

senza, cioè, alcun carattere di separazione tra comandi ed istruzioni.

Collaborazione dei lettori

La collaborazione dei lettori è gradita, purchè si provveda a inviare **almeno** tre sottoprogrammi per volta, su nastro, disco oppure output di stampante. I listati di routine che non rispettano lo standard adottato non potranno esser presi in considerazione.

Tutti i lavori pubblicati verranno compensati con prodotti della Systems Editoriale (cassette di programmi, libri, abbonamenti, copie arretrate, eccetera).

I listati della Systems Editoriale

Un'elevata percentuale dei nostri lettori è alle prime armi nel mondo dell'informatica e incontra difficoltà nel digitare i programmi da noi pubblicati.

I caratteri "speciali" bianchi su fondo nero (semi-grafici in "reverse") che rappresentano precisi comandi per i computer Commodore sono riportati nel listato di esempio a sinistra così come appaiono digitandoli su video o su stampante, mentre a destra come li rappresentiamo nei nostri listati.

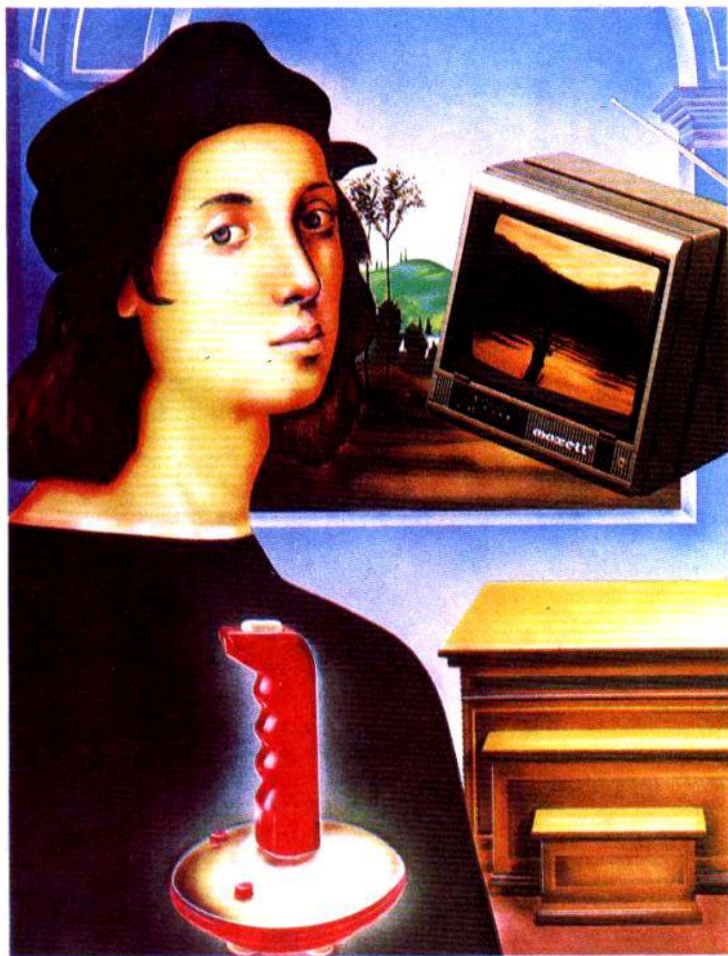
La riga 360, ad esempio, deve così esser interpretata:
dopo aver battuto il carattere di virgolette (") che si ottiene premendo il tasto SHIFT insieme con il tasto 2, è necessario battere il carattere CRSR DOWN (il tasto, cioè, che normalmente sposterebbe il cursore nella cella video sottostante).

Analogamente, nella riga 180 del listato "tradotto" (di destra), il termine [NERO] sta a significare che bisogna utilizzare il carattere speciale del colore nero (tasto CTRL insieme con il tasto 1, vedi listato di sinistra).

Per ricordare in che modo vengono normalmente visualizzati i caratteri speciali, nella seconda parte delle righe di sinistra (dopo i REM) sono riportati i tasti che è necessario premere per ottenere il carattere-comando "speciale".

La Redazione

```
100 REM I CARATTERI SPECIALI
110 REM DEI COMPUTER COMMODORE
120 REM COME APPAIONO NORMALMENTE
130 REM SU VIDEO O SU CARTA.
140 REM (CTRL = TASTO CONTROL)
150 REM (CMDR = TASTO COMMODORE)
160 REM (CRSR = TASTI CURSORE)
170 :
180 PRINT"■":REM CTRL+1 NERO
190 PRINT"■":REM   "   +2 BIANCO
200 PRINT"■":REM   "   +3 ROSSO
210 PRINT"■":REM   "   +4 AZZURRO
220 PRINT"■":REM   "   +5 PORPORA
230 PRINT"■":REM   "   +6 VERDE
240 PRINT"■":REM   "   +7 BLU
250 PRINT"■":REM   "   +8 GIALLO
260 PRINT"■":REM   "   +9 REVERSE ON
270 PRINT"■":REM   "   +0 REVERSE OFF
280 PRINT"■":REM CMDR+1 ARANCIO
290 PRINT"■":REM   "   +2 MARRONE
300 PRINT"■":REM   "   +3 ROSSO CHIARO
310 PRINT"■":REM   "   +4 GRIGIO 1
320 PRINT"■":REM   "   +5 GRIGIO 2
330 PRINT"■":REM   "   +6 VERDE CHIARO
340 PRINT"■":REM   "   +7 BLU CHIARO
350 PRINT"■":REM   "   +8 GRIGIO 3
360 PRINT"■":REM  CRSR IN BASSO
370 PRINT"■":REM  CRSR A DESTRA
380 PRINT"■":REM  CRSR IN ALTO
390 PRINT"■":REM  CRSR SINISTRA
400 PRINT"■":REM  HOME
410 PRINT"■":REM  CANCELLA SCHERMO
420 :
430 REM  ESEMPI DI VISUALIZZAZIONE:
440 PRINT"■■■■■■":REM CANCELLA SCHERMO,
450 :              REM CRSR DWN DUE VOLTE
460 :              REM CRSR DESTRA TRE "
470 :
480 PRINT"■■■■":REM BIANCO,CRSR SINISTRA
490 :          :REM DUE VOLTE E CRSR DWN
500 :          :REM UNA SOLA VOLTA
```

```
100 REM  I CARATTERI
110 REM  SPECIALI: COME
120 REM  VENGONO INDICATI
130 REM  SULLE RIVISTE:
140 REM    COMMODORE
150 REM  E COMMODORE
160 REM  COMPUTER CLUB.
170 :
180 PRINT"[NERO]"
190 PRINT"[BIANCO]"
200 PRINT"[ROSSO]"
210 PRINT"[AZZUR]"
220 PRINT"[VIOLA]"
230 PRINT"[VERDE]"
240 PRINT"[BLEU]"
250 PRINT"[GIALLO]"
260 PRINT"[RVS]"
270 PRINT"[RVOFF]"
280 PRINT"[ARANC]"
290 PRINT"[MARR]"
300 PRINT"[ROSA]"
310 PRINT"[GRIGIO1]"
320 PRINT"[GRIGIO2]"
330 PRINT"[VERDE2]"
340 PRINT"[CELESTE]"
350 PRINT"[GRIGIO3]"
360 PRINT"[DOWN]"
370 PRINT"[RIGHT]"
380 PRINT"[UP]"
390 PRINT"[LEFT]"
400 PRINT"[HOME]"
410 PRINT"[CLEAR]"
420 :
430 REM   ESEMPI
440 PRINT"[CLEAR][2 DOWN]
    [4 RIGHT]"
450 :
460 :
470 :
480 PRINT"[BIANCO][2 LEFT
    ][DOWN]"
```

Una tavolozza per il tuo C/128

Un listato interessante per chi intende sfruttare a fondo le capacità grafiche del potente computer della Commodore

di Roberto Marigo

Chiunque passa dal modo 64 a quello 128, si aspetta, giustamente, di trovarsi di fronte ad una macchina più potente, veloce e, in definitiva, più "capace".

Tutto ciò è vero dal momento che il C/128 è un calcolatore più facile da programmare, più veloce nell'eseguire le varie istruzioni e dotato di una quantità di memoria non accessibile in modo 64.

L'utente del 128 che abbia dato uno sguardo almeno una volta al manuale della macchina, sa benissimo che il BASIC 7.0, linguaggio interprete del computer, dispone di comandi piuttosto potenti per la gestione della grafica; spesso, però, lo stesso utente non sa come usarli, nè in quale tipo di applicazione utilizzarli proficuamente.

Questo pubblicato è un esempio di programma in cui tali comandi possano essere impiegati per un risultato soddisfacente, soprattutto considerando il tempo relativamente breve che occorre per digitarlo. Il programma serve per disegnare figure su video, tramite joystick in porta N.2, colorarle e registrarle (sul dischetto) in modo da riutilizzarle, in seguito, in altri programmi.

Se cercate un potente strumento di CAD (Computer Aided Design, cioè grafica computerizzata), il listato di queste pagine non fa per voi; se, invece, desiderate, più modestamente, scoprire come funzionano le istruzioni grafiche del C/128, e divertirvi, digitatelo subito (senza commettere errori!).

Nel programma vengono, infatti, utilizzati quasi tutti i comandi BASIC riguardanti la grafica, e moltissimi dei comandi per gestire gli sprite in modo efficace e veloce.

Cenni sulla struttura del programma

Il programma lavora su uno schermo di 320x200 pixel, ossia in modali-

tà grafica 1: tale modalità permette la gestione di due colori per ciascuna casella di 8x8 pixel (corrispondenti allo spazio occupato, di norma, da un carattere): il colore di sfondo e il colore di primo piano. In questo modo si raggiunge un risultato migliore che non ricorrendo al modo multicolore che, per aumentare il numero di colori disponibili, è costretto a dimezzare la risoluzione a 160x200 pixel.

Tale scelta pone, però, un limite: tentando di disegnare in uno dei 1000 quadrati di 8x8 pixel con un colore differente da quello ivi già presente, quest'ultimo cambierà, assumendo la colorazione dell'ultimo colore selezionato in ordine di tempo raggiungendo, talvolta, risultati inaspettati e sgradevoli.

Per ovviare a questo inconveniente è presente un secondo cursore-sprite che, attivato, si sovrappone costantemente sulla casella in cui sta disegnando il cursore principale. I due cursori sono facilmente distinguibili poichè il primo (sprite N.1) è a forma di croce, il cui centro corrisponde al pixel in cui si disegna; il secondo, invece, è un quadrato pieno di 8x8 pixel.

Il secondo cursore delimita, istante per istante, la casella in cui si sta disegnando, ricordando, in tal modo, che nella zona può esservi soltanto un colore. Il lettore, di conseguenza, presterà attenzione a non superare, col joystick, la zona delimitata per non "invadere" zone limitrofe. Dopo alcune prove, con i due cursori attivati e spostandosi di un pixel per volta, si può evitare di "pasticciare" una zona già colorata passandovi sopra una linea di diverso colore.

Nel disegno possono essere visualizzati anche caratteri alfanumerici al fine di dare una intestazione al disegno stesso oppure per scrivervi un messaggio. Tale funzione è esplicata dal comando BASIC "TEXT", che scrive nella pagina grafica una determinata stringa in una posizione corrispondente a una casella di 8x8 sulla normale matrice video di 40 colonne per 25 righe. Anche in questa occasione non potrete fare a meno di apprezzare l'utilità del secondo cursore, che indicherà costantemente, se opportunamente attivato, l'esatta posizione in cui verrà visualizzata la prima lettera del messaggio.

Il programma permette l'utilizzo dei 16 colori disponibili sul 128; per cambiare il colore di scrittura corrente è sufficiente la pressione del tasto "C"; il bordo assume così il colore corrente di scrittura. E' da ricordare che ogni disegno o scrittura di testo avverrà sempre nel colore corrente, o se preferite, nel colore del bordo.

I comandi disponibili

Color

Seleziona il colore. Premendo il tasto "C" il colore corrente cambia ed è possibile verificarlo dal fatto che il bordo dello schermo assume la nuova colorazione.

Fast

Cambia la velocità: premendo "F" la velocità del cursore si dimezza o si raddoppia. La sua funzione è di permettere di disegnare con una maggiore precisione quando se ne presenti il bisogno. Premendo nuovamente "F" si ritorna al modo normale.

Premendo i tasti più **(+)** oppure meno **(-)** il cursore aumenta o diminuisce di velocità. Il valore di default è 1 con il modo fast inserito; con il modo slow (nuova pressione del tasto "F" se si è in modalità fast) si ottiene la minor velocità possibile; premendo il tasto "+" per dieci volte di seguito e inserendo la modalità fast si ottiene la massima velocità disponibile.

Nota: la modalità fast o slow non coicide con gli analoghi comandi BASIC: si limita semplicemente a raddoppiare o dimezzare il numero di pixel tracciati. Non è possibile visualizzare la modalità corrente o la velocità: tuttavia la regolazione della velocità risulta agevole e efficace.

Draw mode

Premendo "D" il computer traccia un punto, corrispondente al centro del cursore principale, e continua a tracciarlo fino alla successiva pressione del tasto "D". Al momento del RUN ci si trova in modalità di disegno.

Save/Load

Premendo il tasto "S" è possibile salvare la schermata grafica visualizzata. Si digita il nome del file che viene salvato come file programma, ossia utilizzabile da un altro programma semplicemente settando il modo grafico e caricandolo con il comando BLOAD.

Premendo altresì il comando "L" si può caricare da disco una schermata precendentemente salvata.

Con il comando "L" è possibile esaminare la directory del dischetto rispondendo "D" alla domanda "Filename?".

Attivazione dei cursori

Ambedue i cursori possono essere attivaati o disattivati:
X attiva/disattiva il cursore principale
A attiva/disattiva il secondo cursore

Width (larghezza)

Le righe verticali possono essere allargate o ristrette: premendo "W", appena attivato il programma, le linee verticali risulteranno di ampiezza doppia (2 pixel). Non premendo (oppure premendo il tasto "W" un numero di volte pari) le linee verticali verranno formate da un solo pixel.

Box

Tramite questa opzione si può disegnare un quadrato o un rettangolo: premendo una volta "B", quando il cursore-croce è posizionato nell'angolo superiore sinistro del rettangolo (o quadrato) da disegnare, viene tracciato un punto come pro-memoria. Premendo una seconda volta "B", dopo essersi posizionati sull'angolo inferiore destro, il quadrato (o il rettangolo) viene visualizzato.

Ovals/Circles

Tramite questa opzione si può disegnare un cerchio oppure un'ellisse: premendo "O" sul centro del cerchio si visualizzerà una croce identica al cursore principale al centro del cerchio per ricordare la posizione del centro; in seguito, muovendo il joystick a destra o a sinistra, si determina il raggio; per ottenere un cerchio premete due volte "O" sull'estremo

del raggio. Se, invece, volete ottenere un'ellisse, si determina dapprima la lunghezza dell'ellisse spostando il joystick a destra (o a sinistra) e premendo "O" una prima volta; in seguito si stabilisce la sua altezza spostando il joystick in alto (o in basso); premendo nuovamente "O" si ottiene l'ellisse.

Erase

Premendo il tasto "E" si cancella la pagina grafica. Il computer, per prudenza, ne chiede conferma attendendo la pressione del tasto "S"; in caso contrario si ritorna alla "normalità".

Paint

Premendo il tasto "P" si colora l'area presente attorno al cursore fino a trovare una delimitazione. Attenzione, perchè se il computer non trova un limite, o se trova un punto di interruzione in una figura, il colore "uscirà" dalla figura, sporcando irrimediabilmente il disegno.

Testo

Il tasto "T" permette di inserire un testo nella pagina grafica facendone comparire il primo carattere esattamente ove indicato dal secondo cursore. Premendo il tasto "T" il computer chiede la stringa testo con un semplice INPUT (non superare le due linee di dati) e poi, premendo il tasto Return, la visualizza sulla pagina grafica.

Posizionamento

Premendo uno dei tasti qui indicati il cursore si posiziona in alcuni punti particolari, e precisamente:

(freccia a sinistra) nel centro del video
1 in alto a sinistra
2 in alto a destra
3 in basso a sinistra
4 in basso a destra

Menù di cancellazione

Data la difficoltà di cancellare un cerchio già tracciato, questa routine svolge il lavoro al vostro posto: premendo "M" il computer chiede quale dei cerchi disegnati deve essere cancellato, numerando i cerchi presenti sullo schermo con valori compresi tra zero e 99. Non si possono, quindi, tracciare più di 100 cerchi, o meglio, disegnandone altri, l'ultimo cerchio tracciato non potrà più essere cancellato da programma, ma solo manualmente. Successivamente il computer farà lampeggiare il cerchio indicato (cancellandolo e ridisegnandolo subito dopo) e attenderà la pressione del tasto "S" come conferma. Alla pressione di qualunque altro tasto, tornerà alla normalità, ignorando il comando.

```
100 REM DISEGNATORE PER C/128 IN MODO 40 COLONNE
110 REM BY ROBERTO MARIGO
120 REM JOYSTICK IN PORTA N.2
130 KEY1,CHR$(133):KEY7,CHR$(136)
140 FAST:CLR:GRAPHIC1,1:COLOR1,1:COLOR1,1:COLOR4,1:DIMCC%(99,3):REM INIZIALIZZA
PAGINA GRAFICA
150 DRAW1,10,6TO10,9:DRAW1,5,11TO8,11:DRAW1,10,13TO10,16:DRAW1,12,11TO15,11:REM
DISEGNA LA CROCE CURSORE
160 SSHAPEA$,1,1,22,22:SPRSAVA$,1:SPRSAVA$,2:CC=0:T=0:C=1:F=1:X=99:Y=160:XX=X:YY
=Y:T2=1:IM=1:REM E LA METTE NELLA STRINGA A$ E POI NELLO SPRITE #1
170 GRAPHIC1,1:BOX1,7,8,14,15,0,1:REM DISEGNA UN QUADRATO PIENO:
180 SSHAPEA$,1,1,22,22:SPRSAVA$,3:SPRITE3,0,16:SPRITE2,0,1:REM E LA METTE NELLA
STRINGA A$ E POI NELLO SPRITE #3
190 SLOW:SPRITE1,1,1:COLOR0,2:GRAPHIC1,1
200 J=JOY(2):MOVSPR1,Y+15,X+40:X2=INT(X/8)*8+43:Y2=INT(Y/8)*8+18:MOVSPR3,Y2,X2:X
X=X:YY=Y:REM CONTROLLA JOYSTICK E MUOVI SPRITE
210 A$="":GETA$:IFA$="F"THENF=-F+1:GOTO200:REM CONTROLLA TASTIERA
220 IFF=1ANDA$=""THEN490:ELSE240
230 IFA$="D"THEND=-D+1:GOTO200
240 IFA$="C"THENBEGIN:ELSE290
250 C=C+1:IFC>16THENC=1:ELSEC=C
260 COLOR1,C:COLOR4,C
270 BEND
280 IFA$="C"THENCOLOR1,C:COLOR4,C
290 IFA$="S"ORA$="L"THEN720
300 IFA$="X"ANDT=0THENSPRITE1,0:T=1:GOTO200:REM SPEGNI IL CURSORE
310 IFA$="A"ANDT2=0THENSPRITE3,0:T2=1:GOTO200:REM SPEGNI IL CURSORE 2
320 IFA$="O"ANDRAD=0ANDBOT=0THEN770
330 IFBOT=1ANDA$="O"THEN790
340 IFRAD=1ANDA$="O"THEN780
350 IFA$="X"ANDT=1THENSPRITE1,1:T=0:GOTO200:REM ACCENDI IL CURSORE
360 IFA$="A"ANDT2=1THENSPRITE3,1:T2=0:GOTO200:REM ACCENDI IL CURSORE 2
370 IFA$="W"ANDW=0THENWIDTH2:W=1:GOTO200:REM SETTA DIMENSIONE RIGHE VERTICALI
380 IFA$="W"ANDW=1THENWIDTH1:W=0:GOTO200:REM LARGHEZZA LINEA
390 IFA$="B"ANDB=1THEN840:REM DISEGNO DI UN RETTANGOLO/QUADRATO
400 IFA$="B"ANDB=0THEN830
```

```
410 IFA$="E"THENGOSUB910:GOTO200:REM CANCELLAZIONE SCHERMO
420 IFA$="M"THEN860:REM MENU CANCELLAZIONE CERCHI/ELLISSI
430 IFA$="P"THEN710:REM PAINT: COLORAZIONE DI ZONE DELIMITATE
440 IFA$="T"THEN850:REM SCRITTURA DI CARATTERI ALFANUMERICI
450 IFA$="D"THEND=-D+1:GOTO200:REM ATTIVA/DISATTIVA LA MODALITA' DI DISEGNO A MA
NO LIBERA
460 IFA$="+"THENIM=IM+1:IFIM>12THENIM=12:GOTO200:REM AUMENTA VELOCITA' CURSORE
470 IFA$="-"THENIM=IM-1:IFIM<1THENIM=1:GOTO200:REM DIMINUISCI VELOCITA' CURSORE
480 IFA$="←"THENX=99:Y=160:GOTO200:REM METTI IL CURSORE NELLE VARIE POSIZIONI
490 IFA$="1"THENX=0:Y=0:GOTO200
500 IFA$="2"THENX=0:Y=319:GOTO200
510 IFA$="3"THENX=199:Y=0:GOTO200
520 IFA$="4"THENX=199:Y=319:GOTO200
530 IFJ=0THEN200
540 ONJGOSUB580,570,610,600,630,650,690,680:REM GESTISCE POSIZIONI DEL JOYSTICK
550 IFD=1THEN200
560 DRAW1,Y,XTOYY,XX:GOTO200
570 Y=Y+IM:IFY>319THENY=319:REM LIMITE MASSIMO PER DISEGNO IN ORIZZONTALE
580 X=X-IM:IFX<0THENX=0:REM LIMITE MASSIMO PER BORDO SINISTRO
590 RETURN
600 X=X+IM:IFX>199THENX=199:REM LIMITE MASSIMO PER BORDO INFERIORE
610 Y=Y+IM:IFY>319THENY=319
620 RETURN
630 X=X+IM:IFX>199THENX=199
640 RETURN
650 X=X+IM:IFX>199THENX=199
660 Y=Y-IM:IFY<0THENY=0:REM LIMITE MASSIMO PER BORDO SUPERIORE
670 RETURN
680 X=X-IM:IFX<0THENX=0
690 Y=Y-IM:IFY<0THENY=0
700 RETURN:REM TERMINE ROUTINE GESTIONE JOYSTICK
710 D=1:PAINT1,Y,X:GOTO200:REM ROUTINE DI RIEMPIMENTO DI AREE
720 GRAPHIC0,1:COLOR0,2:COLOR5,1:SPRITE1,0:REM ROUTINE SALVATAGGIO/CARICAMENTO
730 INPUT"FILENAME: ";N$:IFA$="L"THEN750
740 PRINT"▮▮SAVING "N$:BSAVE(N$),B0,P7168TOP16384:GRAPHIC1:SPRITE1,1:GOTO200
750 IFN$="D"THENCATALOG:GOTO730
760 PRINT"▮▮LOADING ";N$:BLOAD(N$),B0,P7168:T=1:D=1:GRAPHIC1:GOTO200
770 XC=X:YC=Y:RAD=1:SPRITE2,1:MOVSPR2,YC+15,XC+40:GOTO200:REM ROUTINE DI CIRCLE
ED ELLISSI
780 R=ABS(YC-Y):BOT=1:RAD=0:GOTO200
790 IFX=XCTHENBR=R:ELSEBR=ABS(XC-X)
800 CIRCLE1,YC,XC,R,BR:BOT=0:SPRITE2,0:CC%(CC,0)=YC:CC%(CC,1)=XC:CC%(CC,2)=R:CC%
(CC,3)=BR
810 CC=CC+1:IFCC=100THENCC=99
820 GOTO200
830 B=1:D=1:X1=X:Y1=Y:DRAW1,Y,X:GOTO200:REM ROUTINE DI DISEGNO DI QUADRATI E RET
TANGOLI
840 X2=X:Y2=Y:BOX1,Y1,X1,Y2,X2:B=0:GOTO200
850 XT=INT(X/8):YT=INT(Y/8):GRAPHIC0,1:INPUT"TESTO :";T$:GRAPHIC1:CHAR1,YT,XT,T$
:GOTO200:REM ROUTINE DI TESTO
860 GRAPHIC0,1:INPUT"▮▮▮▮▮▮NUMERO CERCHIO (0-99) ";CX:IFCX>99ORCX<0THEN860
870 GRAPHIC1,0:CIRCLE0,CC%(CX,0),CC%(CX,1),CC%(CX,2),CC%(CX,3)
880 CIRCLE1,CC%(CX,0),CC%(CX,1),CC%(CX,2),CC%(CX,3)
890 GETKEYA$:IFA$<>"S"THEN200
900 CIRCLE0,CC%(CX,0),CC%(CX,1),CC%(CX,2),CC%(CX,3):GOTO200
910 GRAPHIC0,1:PRINT"CANCELLO TUTTO - SICURO (S/N) ?"
920 GETKEYX$:IFX$<>"S"THEN940
930 GRAPHIC1,1:X=99:Y=160:C=1:COLOR4,C:COLOR1,C:GOTO200:REM CANCELLA PAGINA GRAF
ICA
940 GRAPHIC1,0:RETURN
950 END
```

Uno schermo più allegro

Tre nuove routine per il tuo sistema personale; ed una versione più efficiente di altre due già pubblicate in precedenza

a cura di Alessandro de Simone

Lampeggio righe (per C/64) (22935/23000)

Il "lampeggio" di un messaggio sullo schermo si ottiene, con questa routine, in due formati diversi: ripetendo alternativamente il messaggio nei modi di stampa "Rvs/Rvoff", oppure alternando il colore della stampa da bianco a blu; assumendo che quest'ultimo sia il colore del fondo, si avrà un effetto "appare/scompare".

Per ottenere il lampeggio si deve far precedere la stampa di ciascun messaggio da

SYS XXXXX,A,B

in cui la variabile A e quella B possono variare nei rispettivi intervalli: A=0-2; B=0-7. La routine esegue dapprima l'istruzione JSR$B7F1 che "salta" la virgola e preleva dal testo il primo parametro (A). Se questo è uguale a 2, vengono posti nei registri $FB-$FC i codici ASCII corrispondenti a "bianco/blu"; se invece vale 0 oppure 1, negli stessi registri vengono posti i codici di "Rvs/Rvoff".

In maniera analoga viene prelevato il secondo parametro (B) che determina la frequenza del lampeggio con un valore compreso tra 0 e 7: utilizzando questo dato in combinazione con il byte inferiore dell'orologio ($A2), viene fissato il periodo con cui la routine stampa, in uscita (mediante JSR$FFD2), il contenuto del registro $FB o quello del registro $FC.

Tale periodo è di 1/60 di secondo per B=0, e raddoppia per ogni unità in più fino a circa 2 secondi per B=7. Per il lampeggio Rvs/Rvoff, inoltre, il tipo di stampa viene invertito a seconda che A valga zero oppure uno: ciò permette di far lampeggiare più linee in controfase, eventualmente con periodi diversi.

Naturalmente sarà necessario, alla fine di ogni messaggio o serie di messaggi, inserire tanti "cursor up" quante sono le linee, in modo da riposizionare il cursore correttamente: un successivo GET di attesa ripete l'intero ciclo dall'inizio finchè non si preme un tasto.

Subito dopo arresta il programma richiamando con un "loop" la subroutine GETIN del Kernal (JSR $FFEA), finchè quest'ultima non carica nell'accumulatore un valore diverso da zero (cioè tasto premuto): tale valore, infine, viene inserito nel registro della variabile-stringa prescelta, prima di tornare al Basic.

```
1000 PRINTCHR$(147)"QUESTA ROUTI
     NE CONSENTE DI FAR LAMPEGGI
     ARE DEI MESSAGGI ";
1010 PRINT"SULLO SCHERMO NEL MOD
     O  RVS/RVOFF O NEL MODO ON/
     OFF (SFONDO BLU)"
1020 RETURN
1030 DATA 32,241,183,224,2,208,1
     4,169
1040 DATA 0,133,253,169,5,133,25
     1,169
1050 DATA 31,133,252,208,10,169,
     18,133
1060 DATA 251,169,146,133,252,13
     4,253
1070 DATA 32,241,183,169,1,224,0
     ,240
1080 DATA 6,10,6,253,202,208,250
     ,120
1090 DATA 37,162,88,69,253,208,6
     ,165
1100 DATA 251,32,210,255,96,165,
     252,32
1110 DATA 210,255,96
1120 DATA -1,9465
```

```
100 REM DEMO LAMPEGGIO MESSAGGI
110 REM BY ROBERTO MORASSI
120 PRINTCHR$(147)
125 :REM 22935=INDIRIZZO UTILIZ
    ZATO SU CCC
130 SYS22935,0,4:PRINT"PRIMO LA
    MPEGGIO"
140 SYS22935,1,3:PRINT"SECONDO
    LAMPEGGIO"CHR$(145)CHR$(145
    )
```

```
150 GET A$:IF A$="" THEN 130
160 PRINT:PRINT
170 SYS22935,2,5:PRINT"TERZO LA
    MPEGGIO"CHR$(145)
180 GET A$:IF A$="" THEN 170
190 POKE 53281,0:PRINTCHR$(5):R
    EM SCHERMO NERO E CURSORE B
    IANCO
999 END
```

Ciclo di attesa (per C/64) (23001/23034)

Il ben noto "loop" di attesa del tipo

10 GET A$:IF A$="" THEN 10

che, in Basic, arresta il programma finchè non si preme un tasto e poi assegna il CHR$ di quest'ultimo alla variabile A, può essere sostituito con una SYS che punti a questa routine.

E' però necessario che la variabile-stringa in questione (A$, o un'altra qualunque) venga "dichiarata" prima di qualsiasi altra stringa all'inizio del programma con un'istruzione del tipo: A$="A".

La routine proposta utilizza, infatti, i puntatori di inizio variabili ($2D-$2E) per individuare il registro in cui è contenuto il valore della variabile-stringa, e memorizza tale registro (nella forma byte basso, byte alto) in $FD-$FE.

```
1000 PRINTCHR$(147)"QUESTA ROUTI
     NE ARRESTA IL PROGRAMMA FIN
     O ALLA ";
1010 PRINT"PRESSIONE DI UN TASTO
     , E ASSEGNA    ";
1020 PRINT"QUEST'ULTIMO ALLA PRI
     MA VARIABILE IN MEMORIA"
1030 RETURN
1040 DATA 165,45,24,105,3,133,25
     1,165
1050 DATA 46,105,0,133,252,160,0
     ,177
1060 DATA 251,133,253,200,177,25
     1,133
1070 DATA 254,32,228,255,240,251
     ,160
1080 DATA 0,145,253,96
1090 DATA -1,5076
```

```
100 REM DEMO PER CICLO DI ATTES
    A
110 REM BY ROBERTO MORASSI
120 A$="L":PRINTCHR$(147)"PREMI
     UN TASTO"
130 SYS23001:REM INDIRIZZO UTIL
    IZZATO SU CCC
140 PRINT"HAI PREMUTO: "A$
999 END
```

Riempi-schermo (per C/64) (23035/23085)

Questa routine riempie istantaneamente lo schermo con uno stesso carattere colorandolo, contemporaneamente, con un colore prefissato: si possono così ottenere, soprattutto usando i caratteri semi grafici, effetti spettacolari.

Si usa, come al solito, il comando SYS XXXXX,A,B in cui le variabili A e B sono, rispettivamente, i codici-schermo (e non ASCII!) del carattere e del colore prescelti.

La routine preleva questi due valori con il solito JSR$B1F7, dopo aver predisposto gli opportuni puntatori (da $FB a $FF) all'inizio della memoria di schermo e della memoria-colore ($D800).

Un doppio ciclo provvede poi ad inserire, alternativamente, il carattere ed il colore in quattro pagine consecutive delle rispettive zone di memoria. Tenete presente che i registri-schermo riempiti non sono 1000 bensì 1024, e comprendono quindi anche gli otto puntatori di sprite che si trovano "dopo" la memoria di schermo.

Ciò si è reso necessario per ottimizzare la stessa routine che, altrimenti, avrebbe richiesto un maggior numero di byte.

```
1000 PRINTCHR$(147)"QUESTA ROUTI
     NE RIEMPIE LO SCHERMO CON U
     N UNICO CARATTERE ";
1010 PRINT"E UN UNICO COLORE"
1020 RETURN
1030 DATA 173,136,2,133,252,198,
     252,169
1040 DATA 0,133,251,133,253,168,
     169,215
1050 DATA 133,254,169,4,133,255,
     32,241
```

```
1060 DATA 183,138,72,32,241,183,
     104,230
1070 DATA 252,230,254,72,138,145
     ,253
1080 DATA 170,104,145,251,200,20
     8,245
1090 DATA 198,255,208,237,96
1100 DATA -1,8702
```

```
100 REM DIMOSTRATIVO DELLA ROUT
    INE EMPISCHERMO
110 REM BY ROBERTO MORASSI
115 REM 23035=INDIRIZZO UTILIZZ
    ATO SU CCC
120 SYS23035,1,0:FOR X=0 TO 100
    0:NEXT
130 SYS23035,91,3:FOR X=0 TO 10
    00:NEXT
140 SYS23035,86,1:FOR X=0 TO 10
    00:NEXT
999 END
```

Print At
(Vedi CCC N.35)

Sul N.35 di Commodore Computer Club è già stata pubblicata una routine in L.M. che provvede a simulare il comando PRINT AT che, nel Basic V.2, è assente.

Il motivo per cui ne proponiamo una seconda versione risiede nel fatto che la rubrica "Nuovo Sistema" vuole rappresentare una palestra in cui gli stessi lettori, lavorando su materiale già pubblicato, pervengono a risultati sempre migliori oppure, semplicemente, ad impostazioni, e risoluzioni, diverse di uno stesso problema.

Naturalmente proponiamo la routine non numerata nel modo "convenzionale" poichè potrebbe generare confusione in sede di compattamento: al lettore, dunque, la decisione di utilizzarla come meglio ritiene opportuno, tenendo presente che, come tutte le routine di questa rubrica, è rilocabile ovunque in memoria Ram.

Con questa brevissima routine viene simulata l'istruzione PRINT AT, che stampa una qualunque stringa a partire da un punto prefissato dello schermo. La sintassi da usare è la solita...

SYS XXXXX,Y,X,A$

...in cui Y e X sono, rispettivamente, la posizione del cursore sulla riga (da 0 a 39) e il numero della riga stessa (da 0 a 24); A$ è la variabile-stringa (o l'effettiva stringa) che deve essere stampata a partire dalla posizione (Y,X).

La routine preleva i parametri Y e X dal testo, con due istruzioni JSR$B1F7, e li dispone nei corrispondenti registri del micro: richiama poi la subroutine PLOT del Kernal (JSR $FFF0) che sposta il cursore nel punto voluto, "salta" la virgola con JSR$AEFD, e passa infine il controllo, con un JMP$AAA4, alla subroutine dell'interprete che legge la stringa dal testo e la stampa sullo schermo.

Da sottolineare che se la stringa è formata da N spazi vuoti, si avrà come effetto la cancellazione di N caratteri a partire da (Y,X).

```
100 REM PRINT AT (VERSIONE 2) I
    N LINGUAGGIO MACCHINA
110 REM BY ROBERTO MORASSI
111 :
115 REM LA VERSIONE N.1 E' STAT
    A PUVBBLICATA SU CCC N.35
116 :
117 REM DIGITARE GLI OPPORTUNI
    PARAMETRI AL POSTO DI XXXXX
120 PRINTCHR$(147)
130 SYSXXXXX,10,5,"*STAMPA DA (
    10,5)"
140 SYSXXXXX,20,15,"*STAMPA DA
    (20,15)"
150 SYSXXXXX,2,21,"*STAMPA DA (
    2,21)"
999 END
1000 PRINTCHR$(147)"QUESTA ROUTI
     NE CONSENTE DI STAMPARE UNA
      STRINGA ";
1010 PRINT"IN UN DETERMINATO  PU
     NTO DELLO  SCHERMO"
1020 RETURN
1030 DATA 32,241,183,138,72,32,2
     41,183
1040 DATA 104,168,24,32,240,255,
     32,253
1050 DATA 174,76,164,170
1060 DATA -1,2814
```

Pausa (Vedi CCC N.36)

Anche questa è una versione diversa di una routine già comparsa sulla nostra rivista.

Per inserire una pausa di durata variabile in un programma Basic si ricorre di regola ad un ciclo "a vuoto" del tipo:

FOR X=1 TO 1000:NEXT

La routine pubblicata in queste pagine "ferma" il programma per tempi variabili da mezzo secondo a circa due minuti: viene attivata con SYS XXXXX,A dove A (1-255) è il numero di unità di tempo lunghe mezzo secondo ciascuna; ad esempio, con SYS XXXXX,20 il programma riparte dopo 10 secondi esatti.

La routine preleva il parametro A (saltando la virgola) con un JSR$B1F7, e lo inserisce nel registro $FB. Richiama poi due subroutine del Kernal, e cioè SETTIM (JSR$FFDB) per azzerare l'orologio interno, e RDTIM (JSR$FFDE).

Quest'ultima viene richiamata in ciclo continuo finchè l'accumulatore (in cui viene letto il byte più basso del clock) non contiene il valore 30: poichè ogni "scatto" dell'orologio è di 1/60 di secondo, il ciclo si interrompe dopo 1/2 secondo. A questo punto viene decrementato di 1 il registro $FB, e l'intero ciclo viene ripetuto finchè questo registro non si azzera.

```
100 REM PAUSA: VERSIONE 2
110 REM BY ROBERTO MORASSI
111 :
115 REM LA VERSIONE 1 E' STATA
    PUBBLICATA SU CCC N.36
116 REM DIGITARE L'OPPORTUNO PA
    RAMETRO AL POSTO DI XXXXX E
     YYYYY
117 :
120 PRINTCHR$(147)"PAUSA DI 5 S
    ECONDI":SYSXXXXX,10
130 PRINT"PAUSA DI 10 SECONDI":
    SYSYYYYY,20
999 END
1000 PRINTCHR$(147)"QUESTA ROUTI
    NE ARRESTA IL PROGRAMMA PER
     UN ";
1010 PRINT"TEMPO PREFISSATO, IN
    UNITA' DI 0.5 SECONDI"
1020 RETURN
1030 DATA 32,241,183,134,251,169
    ,0,32
1040 DATA 219,255,32,222,255,201
    ,30,208
1050 DATA 249,198,251,208,240,96
1060 DATA -1,3706
```

Per chi inizia

Ricordiamo, ai nuovi lettori che ci leggono per la prima volta, che la nostra rivista propone su ogni numero un gruppo di routine in Linguaggio Macchina, per il Commodore 64, interamente rilocabili. Ciò significa che l'utente potrà realizzare una vera e propria enciclopedia in L.M. personalizzata in base alle proprie esigenze.

E' ovvio che le routine proposte, per funzionare adeguatamente, soggiacciono ad alcuni limiti, ben illustrati nei primi numeri di "Nuovo Sistema".

In questa sede ci limitiamo a ricordare che, per evitare malfunzionamenti, è necessario:

- **Fissare il Top di memoria Ram a 20000.**
- **Caricare, servendosi del programma "Caricatore" a suo tempo pubblicato (o analoghi), le routine Basic contenenti le istruzioni Data.**
- **Indicare la prima locazione di memoria in cui si desidera allocare la routine stessa.**
- **Attivare la routine secondo i suggerimenti indicati nei Demo a corredo.**

Mappa della memoria di NUOVO SISTEMA

(Elenco delle routine pubblicate)

Il primo valore indica l'indirizzo di partenza (coincidente con la SYS da impartire), mentre, il secondo, l'ultima locazione contenente l'ultimo dato.
Il numero fra parentesi, invece, si riferisce al numero di C.C.C. in cui sono state pubblicate le routine stesse.

20000/20011 GoTo Calcolato (31)
20012/20049 GoSub Calcolato (31)
20050/20128 Interp A$ (31)
20129/20188 Cambia colore (31)
20189/20245 Scroll Carattere (31)
20246/20302 Cancella caratt. (31)
20303/20445 GoSub Label (32)
20446/20562 GoTo Label (32)
20563/20596 Restore linea (33)
20597/20682 Disk Tool (33)
20683/20775 Directory (33)
20776/20858 Scroll Flag (34)
20859/20914 Deek (34)
20915/20952 Doke (34)
20953/21106 Decim/Esadec (35)
21107/21156 Locate cursor (35)
21157/21260 Beep (35)
21261/21473 Def.Sprite (36)
21474/21839 Sprite tool (36)
21840/21919 Colisione sprite (36)
21929/21962 Pause (36)
21963/22035 Cancella schermo (37)
22036/22174 Effetti sonori (37)
22175/22195 Up Scroll (37)
22196/22357 Right Scroll (37)
22358/22506 Left Scroll (37)
22507/22598 Down Scroll (37)
22599/22710 Comando Mid$ (38)
22711/22781 Comando Instr (38)
22782/22934 Help variabili (38)

(Le routine di questo numero sono opera di Roberto Morassi)

Considerando che i numeri 1, 2 e 7 sono esauriti, vogliate inviarmi i numeri arretrati .... al prezzo di L. 5.000 cadauno per richieste fino a 4 numeri, o di L. 4.000 cadauno per richieste oltre i 4 numeri arretrati, e perciò per un totale di L.............. Sono a conoscenza che i fascioli suddetti non saranno inviati in contrassegno e, pertanto, ho provveduto oggi stesso a versare il canone di L.................. a mezzo c/c postale n. 37952207 intestato a:
Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

GIUDIZIO SUI PROGRAMMI DI QUESTO NUMERO

Ho assegnato un voto da 0 a 10 ai programmi che indico di seguito:

A/ .... Voto ....

B/ .... Voto ....

C/ .... Voto ....

D/ .... Voto ....

PICCOLI ANNUNCI..

....

....

....

....

....

CERCO/OFFRO CONSULENZA..

....

....

....

....

....

INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

COMMODORE COMPUTER CLUB

V.le Famagosta, 75
20142 Milano

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... **Cognome** ....

Via .... **N°** .... **CAP.** .... **Città** ....

Telefono .... **Orario** ....

Quale fascicolo manca alla tua enciclopedia Commodore?

Per ordinare i fascicoli mancanti alla tua collezione di Commodore Computer Club utilizza l'apposita scheda in fondo alla rivista.

Ora anche su disco

MS-DOS
&
GW-BASIC
EMULATOR

TRASFORMA
IL TUO 64
IN UN
PC IBM

"MS-DOS & GW-BASIC emulator" è anche su disco. Per quanti hanno acquistato la versione su cassetta ed inviano la relativa prova d'acquisto, il dischetto è disponibile a lire 15.000 (+ lire 3.000 per spese di spedizione). Non occorre inviare la cassetta nè tantomeno il manualetto di istruzioni. Chi non è in possesso della cassetta può richiedere il disco ed il manualetto al prezzo normale di lire 25.000 (+ lire 3.000 per spese di spedizione).

Per una veloce evasione dell'ordine inviate un assegno bancario o circolare non trasferibile all'ordine della "Systems Editoriale" (V.le Famagosta, 75 - 20142 Milano).

Sempre un passo avanti.

Come gestire senza problemi i file sequenziali

Una trattazione approfondita e ricca di esempi e consigli per evitare grossolani errori nel manipolare i file su disco o cassetta.

di Alessandro de Simone

Una delle principali applicazioni di un computer, come è noto, è quella di manipolare una gran quantità di informazioni di ogni tipo. Se, poi, i dati da tenere a mente superano la disponibilità di memoria RAM, oppure è necessario conservarli per future elaborazioni, diventa indispensabile la possibilità di "scaricarli" su memorie di massa, in modo da renderli disponibili in qualsiasi momento.

Le memorie di massa sono quei particolari dispositivi che permettono di registrare su supporto magnetico le informazioni sotto forma diversa. Scopo del presente inserto è quello, appunto, di guidare il lettore nella comprensione dei vari stadi della registrazione (e successiva lettura) non solo per programmare correttamente ma, soprattutto, per individuare più facilmente eventuali errori che inavvertitamente si dovessero compiere.

Che cosa è un file

La parola inglese FILE significa "Raccolta", "Schedario" ed altri termini, insomma, che richiamano alla mente un insieme ordinato di elementi.

L'elenco telefonico, l'indice di un libro, le varie "voci" di un estratto conto bancario sono altrettanti file che rispettano rigidamente alcune regole necessarie alla decodifica di ciascun elemento dello schedario.

I nominativi (record) appartenenti, ad esempio, ad un qualsiasi elenco telefonico (file), sono organizzati in più parti (campi) ciascuno dei quali ha un preciso significato: il primo è il cognome, il secondo il nome, il terzo l'indirizzo e l'ultimo il numero di telefono.

Vi sono, ovviamente, alcune eccezioni; al posto del cognome e del nome può figurare il solo nome (di una Ditta); vi possono essere più numeri di telefono; può figurare (o meno) il simbolo della segreteria telefonica (Q) o, addirittura, il rinvio ad altra voce (esempio: Pompieri. Vedi: Vigili del Fuoco).

Nonostante le varie eccezioni, alcune delle quali noi stessi ignoriamo fino al momento in cui le incontriamo per la prima volta, la consultazione di un elenco telefonico risulta agevole e chiunque è in grado di rintracciare il numero desiderato.

Un computer, purtroppo, è un po' più rigido della mente umana e se qualcuno gli ha "insegnato" che il terzo campo di un record rappresenta sempre un indirizzo, non è in grado di stabilire che un numero (ivi erroneamente trascritto) non può essere un indirizzo, e prosegue imperterrito la sua elaborazione.

Vedremo tra breve, servendoci di opportuni programmi, il motivo di quest'ultima precisazione.

Simulazione di un file

Noi stessi, senza saperlo, utilizziamo spesso dati sotto forma di file: una qualsiasi conversazione telefonicae rappresenta, ad esempio, uno scambio di dati trasmessi, e ricevuti, parola dopo parola. Per uscire dal quotidiano, tuttavia, ci riferiremo ad un caso particolare.

Pensiamo, infatti, a ciò che accade quando si svolge una conversazione tra radioamatori:

• *Viene manifestata l'intenzione di parlare (Esempio: "Qui SP-10, c'è qualcuno in ascolto?...")*
• *In un modo o in un altro comunichiamo la fine della comunicazione e la richiesta di una risposta (Esempio: "...passo")*
• *In una maniera perfettamente analoga l'eventuale interlocutore risponde, a suo tempo, alla domanda (Esempio: Qui Hg-45 ricevuto messaggio da SP-10. Passo)*
• *La conversazione, poi, continua fino al termine (di solito un "Passo e chiudo").*

Bene, il computer richiede esattamente la stessa procedura, tanto è vero che lo scambio di dati tra calcolatore e periferica (registratore o drive) viene indicata con il temine Handshaking che, letteralmente, vuol dire "Stringersi la mano".

Prima di adoperare i "veri" file, come li utilizza un computer, possiamo quindi riconoscere una perfetta analogia tra il colloquio che intercorre tra due radioamatori e quello che si stabilisce tra computer e memoria di massa:

• *Si apre un file: si esplicita cioè la necessità di iniziare una conversazione.*
• *Si scrivono, sul file prima "aperto", le informazioni che si desidera inviare: si manifesta in modo esplicito il desiderio di mandare (e non di ricevere) informazioni.*
• *Si chiude il file dopo aver inviato l'ultimo dato: si invia un segnale che comunica il termine di tutte le operazioni iniziate.*

Prima di parlare di argomenti più esplicitamente informatici, proseguiamo la nostra "simulazione" e cerchiamo di capire gli inconvenienti che possono capitare se non si rispettano le semplici regole cui accennavamo.

Se, infatti, non "si apre" un file, l'eventuale radioamatore in ascolto non può riconoscere chi desidera parlare nè, tantomeno, se è proprio lui la persona invitata a conversare. Ne consegue che è indispensabile che la prima oparazione da compiere

sia quella di dare il segnale di inizio operazioni (aprire un file).

Se, per errore, si aprisse (prima di chiuderlo) lo stesso file, si genererebbe confusione: il radioamatore chiamato, ad un secondo messaggio del tipo "C'è qualcuno in ascolto?" sarebbe indotto, quanto meno, a credere che la sua prima risposta non sia stata ascoltata. Morale: mai aprire due volte lo stesso file.

Se, poi, si premesse il pulsante di trasmissione della radio mentre si sta ascoltando (oppure, viceversa, quello di ricezione mentre si sta trasmettendo), verrebbe ancora generata confusione, se non altro perchè non si riuscirebbe a ricevere (a trasmettere). Ne consegue che se un file è stato aperto per trasmettere (PRINT) bisogna evitare di compiere operazioni tipiche di ricezione (INPUT) e viceversa.

Gli errori logici

Ora dobbiamo abbandonare l'analogia con le trasmissioni via radio perchè, normalmente, non capita spesso, almeno nelle applicazioni più semplici, di aprire contemporaneamente due file, uno per leggere e l'altro per scrivere.

Capita, invece, di scrivere un file, che rappresenta il risultato dell'elaborazione di un programma e, in seguito, leggere gli stessi dati mediante un altro programma, totalmente diverso dal precedente, pur se, come vedremo, "simmetrico".

Supponiamo di disegnare, sullo schermo in alta risoluzione, una funzione matematica il cui completo tracciamento richieda un'ora intera.

Se abbiamo la necessità di far vedere, nel corso di una conferenza, tale disegno, ed altri grafici che hanno richiesto tempi analoghi, sarebbe davvero noioso aspettare delle ore prima che si compia il disegno. E' molto più comodo, quindi, memorizzare le schermate che appaiono a mano a mano, in modo da richiamarle una alla volta in pochi secondi.

Potremo, quindi, realizzare due programmi: il primo, incaricato di generare (e memorizzare su supporto magnetico) i gra-

fici, ed il secondo in grado di caricarli e visualizzarli.

Potremo far girare il primo programma durante la notte in modo da ritrovare, al mattino successivo, una diecina di disegni, pronti da visualizzare nell'arco di un tempo brevissimo.

Naturalmente bisogna prestare la massima attenzione affinchè i due programmi siano perfettamente simmetrici.

La simmetria

Supponiamo, ora, di aver sviluppato un programma matematico tale che richieda il raggio (R) di una cerchio e fornisca, in risposta, tre dati: il diametro (2*R), l'area del cerchio (3.14*R*R) e la misura della circonferenza (2*3.14*R).

Se questi stessi dati sono memorizzati su nastro (o disco) magnetico per esser manipolati in seguito, è indispensabile che il programmatore sappia con la massima precisione l'ordine con cui sono stati memorizzati.

Consideriamo, infatti, il caso in cui si compia l'elaborazione relativa non ad uno solo, ma ad un centinaio di cerchi, ognuno dei quali genera i tre dati prima visti. Volendo determinare, con un secondo programma che agisse sui dati già memorizzati, la lunghezza dell'arco di circonferenza relativa all'angolo di 45 gradi, sarà sufficiente individuare, in ciascuna terna, la lunghezza della circonferenza e impostare la proporzione...

"Se l'intera circonferenza corrisponde a 360 gradi, quanto misurerà l'arco corrispondente a 45 gradi?"

...che, in termini simbolici, si esprime anche con...

"Lunghezza sta a 360 come X sta a 45"

... e, denominata LU la misura della circonferenza, significa risolvere la semplice equazione:

X=LU*45/360

Sarà quindi necessario scrivere un programma che compia le seguenti operazioni:

• *Rintracciare il file registrato con il programma prima visto che contiene, lo ripetiamo, trecento dati relativi, a gruppi di tre, al diametro, all'area del cerchio e alla circonferenza di ciascun raggio calcolato.*

• *Individuare, all'interno dello stesso file, i dati che interessano, vale a dire il terzo (prima circonferenza), il sesto (seconda circonferenza), il nono (terza circonferenza) e così via.*

In altre parole è assolutamente indispensabile ricordare la modalità secondo cui sono stati registrati i dati; in caso contrario si possono avere sgradite sorprese.

Pensate, infatti, a ciò che potrebbe succedere se, invece di individuare il terzo, sesto... nono dato, il programma esegua l'elaborazione sul secondo, quinto... ottavo dato: nonostante, anche in quest'ultimo caso, il "modulo" sia tre, verrebbe sottoposta ad elaborazione la misura relativa all'area e non alla circonferenza. Il programma, tuttavia, fornirà egualmente dei risultati numerici che, però, non corrisponderanno assolutamente a ciò che si richiede.

Allo stesso modo, se abbiamo a disposizione un archivio in cui, a gruppi di quattro, sono memorizzati gli abbonati del telefono (Cognome, Nome, Indirizzo, Numero di telefono) una eventuale ricerca di un cognome, impostata erroneamente sul "modulo" due (relativo al nome) porterebbe a risultati non richiesti.

In definitiva: è possibile avere la massima libertà nello scrivere dati su supporto magnetico, a patto di ricordare aon la massima precisione l'ordine con cui sono stati memorizzati, in modo da realizzare un programma che, tenendo conto dello schema con cui sono stati memorizzati i dati, rintracci quelli voluti escludendo soltanto quelli che non interessano.

Non è obbligatorio, ovviamente, rispettare un "modulo", ma è intuitivo che il lavoro di ricerca è facilitato se è possibile basarlo tenendo conto di una impostazione ordinata.

La sintassi dei file

Le periferiche verso cui è possibile inviare dati sono di tre tipi: riceventi, trasmittenti e ricetrasmittenti.

Il video va considerato come una vera e propria periferica verso la quale è possibile, però, il solo invio dei dati. Anche la stampante, da questo punto di vista, è una periferica solo ricevente dal momento che è abilitata solo a ricevere dati da stampare su carta. A pensarci bene, però, una stampante invia, di tanto in tanto, dei segnali al computer: poichè la velocità di invio dati, dal computer alla stampante, è superiore alla velocità di scrittura, la periferica è costretta ad inviare un segnale di "sospensione" fintantochè non abbia esaurito di riversare su carta i caratteri già ricevuti. Nonostante questo, però, la stampante viene considerata una periferica ricevente.

Apparecchi ricetrasmittenti sono, al contrario, i modem (che consentono la ricetrasmissione di dati via telefono) e, per ciò che ci riguarda, il registratore ed il drive.

Per iniziare un "colloquio" è comunque necessario manifestare tale intenzione aprendo un file. Il comando inglese corrispondente ad "apri" è OPEN che va seguito da uno o più parametri che meglio specificano le modalità che si intendono seguire.

Nella forma completa il comando Open va seguito da tre parametri che, in seguito, vedremo più da vicino: il numero di file, il numero di periferica e l'indirizzo secondario.

Il numero di periferica

E' molto importante, per ora, tener conto che per ciò che riguarda il numero di periferica c'è una limitata libertà di scelta.

Con il numero zero, infatti, è possibile riferirsi esclusivamente alla tastiera: certo! La tastiera, a tutti gli effetti, è considerata una vera e propria periferica (solo trasmittente). Trascrivete il seguente programma e, dopo il Run, provate a digitar qualcosa e a premere il tasto Return.

```
100 OPEN 1,0
110 INPUT#1,A$
120 PRINT A$
130 GOTO 110
```

Per "uscire" dall'insolita posizione sarà sufficiente premere i tasti Run/Stop e Restore.

Un'altra periferica, cui è assegnato un numero ben preciso (3), è lo stesso video, periferica tipicamente ricevente.

Aggiungete, al programma di prima, la seguente linea:

```
105 OPEN3,3
```

e modificate la 120 come segue:

```
120 PRINT#3,A$
```

Prestate la massima attenzione al fatto che il comando PRINT, seguito dal carattere di cancelletto (#) non può essere abbreviato con il punto interrogativo (?), pena l'emissione di un "Syntax Error". Sembrerebbe che, con le modifiche apportate, il programma giri allo stesso modo di prima ed infatti... è proprio così! Abbiamo preferito, però, parlare egualmente dell'insolita sintassi, se non altro per ricordare l'abbinamento del video al numero 3 di periferica.

Il numero uno, invece, è riservato al registratore e lo vedremo nei programmi applicativi. Il numero due, in "ambiente" Commodore, era riservato al registratore e nei precedenti modelli (ci riferiamo ai famosi PET) era infatti possibile collegare due registratori al calcolatore. Nei più recenti modelli, invece, dal Vic 20 in poi, è stata eliminata la possibilità di servirsi del secondo registratore ed il riferimento alla periferica numero due sortisce un effetto particolare.

Ci riferiamo ai dispositivi esterni (altri computer, stampanti dotate di altri standard, eccetera) che agiscono con la cosiddetta interfaccia RS-232. Questa si collega al Commodore 64, mediante connettore opportuno, alla porta utente, posat sul retro del calcolatore.

Dal momento che non capiterà spesso di collegare altri computer al vostro C/64, eviteremo di dilungarci sull'argomento, ripromettendoci di ritornarvi in un articolo a parte.

Il messaggio di errore "Device not present Error" viene visualizzato nel caso in cui abbiate dotato il vostro computer di Speed Dos oppure di una cartuccia in grado di velocizzare il drive: molto spesso tali apparecchiature aggiuntive escludono la possibilità di servirsi del nastro cassetta.

Il numero di periferica quattro si riferisce, di norma, alla stampante e tutte le stampanti Commodore hanno come standard tale valore; alcune di esse, tuttavia, posseggono un deviatore che, modificando il numero di periferica (di solito: 5) permettono il collegamento contemporaneo di due stampanti ad uno stesso calcolatore. Il vantaggio di un simile accorgimento è evidente: è possibile inviare una parte di dati ad una delle due stampanti, ed altri dati all'altra in modo da ottenere due tabulati diversi senza esser costretti a cambiare, ad esempio, tipo di carta o tipo di margherita o di sfera.

Il numero 8 è quello di solito riservato al drive e ciò spiega il motivo per cui quasi tutti i listati che è possibile reperire per i computer Commodore posseggono tale valore come numero di periferica. Anche noi, negli esempi che seguiranno, ci riferiremo ai due numeri standard: 1 per il registratore, ed 8 per il disk drive.

Iniziamo a parlare

Dopo aver chiacchierato, in generale, del più e del meno sulle periferiche, inizieremo, ora, a parlare dei file in maniera sistematica. Abbiamo detto che la prima operazione da compiere è l'apertura del file, mediante il comando OPEN. Riferendoci, per ora, al registratore, esaminiamo da vicino la forma sintattica per aprire un file in scrittura:

OPEN FN,DN,SA,"PROVA"

Il termine OPEN lo conosciamo già; esamineremo, quindi, gli altri tre, uno per uno.

Per FN si intende il File Number, vale a dire il numero di file al quale intendiamo riferirci. Non va dimenticato, infatti, che è possibile aprire, contemporaneamente, fina a dieci file sequenziali! E' come se decidessimo di parlare con dieci radioamatori: è intuitivo che non è possibile svolgere dieci colloqui allo stesso momento, ma è indispensabile stabilire, volta per volta, con chi abbiamo intenzione di comunicare.

Il numero di file, insomma, servirà nel corso del collegamento per far capire al computer non solo con quale periferica vogliamo intrattenerci ma anche quale compito svolgere. Potrebbe, infatti, capitare il caso in cui siano stati aperti tre file:

- *il primo con la stampante, che consentirà di riversare su carta, se necessario, i risultati dell'elaborazione.*
- *il secondo con il drive, in lettura, per estrarne dati precedentemente memorizzati.*
- *il terzo, sempre con il drive, per creare un nuovo file che rappresenta una nuova elaborazione del file precedente.*

Tanto per chiarire ulteriormente le idee, diremo che dopo aver assegnato il numero di file, che risulterà inevitabilmente "legato" ad una periferica, dovremo, in seguito, utilizzare il solo FN sia per indicare la periferica stessa (drive piuttosto che stampante), sia per assegnarle un particolare compito (leggere e non scrivere).

Vediamo, ora, alcune forme sintattiche corrette ed altre che, invece, generano errori di vario tipo.

Il solo OPEN genera un Syntax error ed è logico che sia così: il computer non ha il minimo elemento per determinare la periferica che desideriamo utilizzare nè, tantomeno, il modo di adoperarla. Ha senso, però, il semplice comando dotato del solo FN:

OPEN 1

In questo modo si comunica l'intenzione di utilizzare in lettura il registratore a cassette; se avete provato a digitare il comando precedente, noterete la comparsa del messaggio "Press play on tape". Premendo il tasto Run/Stop si interrompe l'atte-

sa (Break Error) nella quale si era messo il computer predisposto alla lettura di un file sequenziale dal nastro. Il comando analogo (OPEN1,1) in cui si esplicita la richiesta di collegarsi con un registratore a cassette produce lo stesso effetto.

Dopo aver ricordato che è possibile usare, per FN, un qualsiasi valore numerico compreso tra zero e 255 (pena l'emissione di un "Illegal quantity error") ci soffermeremo molto brevemente sul secondo parametro, il DN (Device Number, numero di periferica).

Questo, come abbiamo già detto, ha un valore ben preciso, almeno se si utilizzano apparecchiature standard, e viene nominato una sola volta, al momento iniziale della "dichiarazione" OPEN. In seguito, infatti, le tre istruzioni tipiche del colloquio tramite file (PRINT#, INPUT#, GET#) si riferiranno ESCLUSIVAMENTE al numero di file FN che contiene, implicitamente, anche il numero di periferica. Nemmeno con il comando di chiusura della comunicazione (CLOSE) sarà necessario ricordarsi di indicare il device!

Il messaggio di errore derivante dalla cattiva manipolazione del DN è "Device not present Error" dovuto alla mancata accensione della periferica. Il messaggio compare dimenticando di fornire energia elettrica alla stampante ed al drive in quanto è presente, all'interno del computer, una sorta di "sensore" che si accorge della loro assenza.

Se, invece, abbiamo dimenticato di collegare il registratore a cassette oppure di connettere il computer all'interfaccia RS-232, non verrà emesso alcun segnale di errore ed il calcolatore resterà in paziente attesa di ricezione dei dati (se predisposto alla lettura) oppure li invierà (se in scrittura) senza tener conto se c'è qualcuno a riceverli!

Il terzo parametro (SA, Secondary Address, indirizzo secondario) non sempre è necessario ma, una volta dichiarato, rappresenterà un modo di comportarsi ben preciso per la periferica. Nel caso del registratore, il SA può assumere uno dei tre valori 0,1,2. Il primo indica l'imposizione di leggere dati dal nastro (Press play on tape), gli altri due, invece, di scriverli (Press record & play on tape). Prima di dilungarci sulla differenza tra i valori 1 e 2, ricorderemo che le tre forme sintattiche:

OPEN 1
OPEN 1,1
OPEN 1,1,0

rappresentano lo stesso comando, vale a dire predisporre il registratore per la lettura del primo file sequenziale che dovesse essere individuato sul nastro non appena venga premuto il tasto Play.

Mentre per la lettura è possibile omettere non solo il DN ma addirittura il SA, in fase di scrittura ciò non è possibile ed è necessario indicarli tutti e tre. Rimane da chiarire la differenza esistente tra SA=1 oppure 2.

E' quindi opportuno aprire una parentesi per affermare che se un file viene registrato è perchè si ha intenzione, in seguito, di leggerne il contenuto. Nel caso del registratore, come è noto, è possibile registrare tanti di quei dati da riempire l'intero nastro. E' sottintesa anche la possibilità di registrare più file diversi tra loro, di diversa lunghezza. Mentre, però, un dischetto contiene un numero di byte ben preciso (circa 160000), sul quale fa affidamento il computer per sapere se il floppy è "pieno" oppure, semplicemente, per sapere dove leggere o scrivere i dati, lo stesso calcolatore non può sapere a priori la lunghezza della cassetta adoperata nè, tantomeno, sapere se la parte di nastro presente davanti alla testina è quella relativa all'inizio, o meno, della cassetta. Risulta necessario, quindi, introdurre un "qualcosa" per cui il computer possa riconoscere se, nell'esame successivo dei vari file, sia giunto alla fine del nastro oppure se sono presenti altri dati dopo quelli che sta leggendo. Naturalmente tale capacità di discernimento è affidata al programmatore in fase di scrittura del file: se, in fase di scrittura, viene utilizzato l'indirizzo secondario 1, in coda al file verrà apposto automaticamente un segnale EOF (End of file, fine del file) che, grossomodo, significa: "altri dati, oltre questi, possono essere eventualmente presenti sul nastro".

Se, invece, viene utilizzato 2 come valore di SA, verrà apposto un segnate di EOT (End of Tape, fine del nastro) che assume un significato diverso: "non vi sono più dati sul nastro; è necessario sostituirlo per effettuare altre ricerche".

E' ovvio che se il computer, nel ricercare un file, incontra un

EOT invece che un EOF (o viceversa), possono verificarsi malfunzionamenti:

• *il nastro viene fatto scorrere inutilmente alla ricerca di un file che non c'è (presenza di EOF invece di EOT).*
• *il nastro viene fermato nonostante, posti di seguito, siano presenti altri file (presenza di EOT erroneamente inserita al posto di EOF).*

La gestione dei file su cassetta, nonostante sia facilitata dalle varie forme sintattiche disponibili, risulta piuttosto laboriosa e, in pratica, è molto meglio utilizzare un nastro per ogni file rinunciando alla possibilità di registrare sullo stesso supporto più file diversi e rintracciarli via software.
Il terzo parametro è comunque indispensabile per registrare dati con il registratore; risulta altresì vitale per il disk drive (sia in lettura che in scrittura) mentre, con la stampante, assume un significato particolare a seconda del valore scelto. Rinviamo il lettore alla consultazione del manuale della stampante dal momento che il significato del valore di SA cambia a seconda del modello.
Un quarto parametro che abbiamo dimenticato di esaminare è il nome del file. Questo, indispensabile lavorando con il drive, risulta non necessario con il registratore. Se, però, lo si dichiara in fase di lettura, verranno scartati tutti i file, eventualmente rintracciati durante la ricerca, presenti sul nastro. Se, ad esempio, registrate un file con il nome di PIPPO:

OPEN1,1,1,"PIPPO"

e, in seguito, lo desiderate leggere, potrete scegliere tra una delle seguenti forme:

OPEN5 oppure OPEN7,1 oppure OPEN3,1,0 :verrà esaminato il primo file che capita di leggere; se però, prima di PIPPO, è presente un altro file, verrà letto il contenuto di questo. Si noti che, in scrittura, è stato usato, come FN, il valore 1. In fase di successiva lettura il numero di file (FN) può essere totalmente diverso (come, appunto: 5, 7, 3).

OPEN5,1,0,"PIPPO" :in questo caso, volendo indicare il nome esplicitamente, è necessario indicare i tre parametri NF, DN, SA; in caso di omissione viene visualizzato il solito Syntax error. Con tale forma sintattica viene esclusa la possibilità di leggere file non desiderati a patto di non aver assegnato lo stesso nome PIPPO a due file registrati in successione!

Si tenga presente che il file registrato col nome PIPPO viene riconosciuto valido se, in lettura, chiedete l'apertura di un file, dal nome più corto, che però coincida con le prime lettere del file registrato. Se, ad esempio, aprite la lettura nel modo seguente:

OPEN 3,1,0,"PIP"

verrà riconosciuto valido non solo un file di nome PIP, ma anche quelli dotati dei nomi PIPPO, PIPE, PIPPONE; solo i file dotati di nomi come POP, PEPPE ed altri, insomma, con le prime tre lettere diverse da PIP, verranno scartati.

Si tenga presente, infine, che se il file PIPPO è stato registrato con il SA pari a 2, desiderando rintracciare un altro file, registrato su nastro dopo di questo, la ricerca verrà bruscamente interrotta da un messaggio di "Device not present error" a causa del segnale di EOT apposto automaticamente alla fine del file.

Chiusura delle trasmissioni

Dopo aver aperto un file, e aver letto oppure scritto a seconda dei casi, è necessario chiuderlo, operazione questa che, corrispondente al "Passo e chiudo", consente al calcolatore di sistemare correttamente i dati in modo che possano esser letti, in seguito, senza alcuna difficoltà.

La sintassi è semplicissima e si riferisce al numero di file precedentemente aperto. Sarà quindi sufficiente un banale CLOSE1 anche se, in apertura, avete dichirato un comando ben più lungo, come il seguente:

OPEN1,1,2,"PROVA"

E' possibile, ovviamente, anche un comando più dettagliato, come questo che segue:

CLOSE1,1,2,"PROVA"

ma il risultato che si ottiene è perfettamente identico al precedente.

Sarà opportuno, ora, considerare i gravissimi malfunzionamenti che si verificano TUTTE le volte che ci si dimentica di chiudere un file.

I casi che si possono presentare sono due ed ognuno di questi può creare problemi non immediatamente avvertibili. Iniziamo dal più semplice, vale a dire dalla dimenticanza di chiudere un file in lettura.

Se, ad esempio, scriviamo un programma che ci permetta di leggere il contenuto di un file (già scritto in precedenza), l'eventuale omissione del comando di chiusura non crea particolari problemi. esempio:

```
100 OPEN 1,1,0,"PROVA"
......
......
500 END
```

Fa eccezione, invece, il caso in cui, con lo stesso programma, si desideri esaminare un altro file, e si assegni lo stesso numero di FN. In questo caso, infatti, si avrebbe una situazione del genere:

```
100 OPEN 1,1,0,"PROVA"
......
......
300 OPEN 1,1,0,"NUMERI"
......
500 END
```

In questo caso, infatti, un messaggio di "File open error in 300" verrebbe immediatamente visualizzato, interrompendo l'elaborazione. Se invece di assegnare il valore "1" a FN asse-

gnassimo un altro valore, la procedura non verrebbe interrotta e tutto procederebbe per il verso giusto; è bene, però, abituarsi a chiudere un file non appena si smette di servirsene.

Se, comunque, dimenticate aperto qualche file, tenete presente che tutte le volte che digitate RUN oppure aggiungete (o cancellate) una riga Basic, o effettuate una cancellazione delle variabili (CLR), tutti i file aperti fino a quel momento vengono automaticamente chiusi.

Diamo ora uno sguardo a ciò che accade, invece, nel dimenticare aperto un file.

Se ciò accade con la stampante i guai, di norma, non sono gravi, tranne qualche inspiegabile comportamento dovuto all'impostazione di indirizzi secondari.

Grave, invece, risulta la dimenticanza di chiusura nel caso del registratore e gravissima, addirittura, è quella che si manifesta utilizzando il drive. Il perchè è presto detto:

Quando, in fase di lettura, esaminiamo un file alla ricerca di un dato, non sempre è possibile sapere a priori quale sia l'ultimo byte memorizzato. E' il caso, tra i tanti, del file generato da un Word Processor (W/P) vale a dire uno di quei particolari programmi professionali che consentono di memorizzare lettere, circolari e documenti in genere. E' impossibile, in questo caso, sapere quanto sarà lungo il documento che intendiamo leggere ma, fortunatamente, un opportuno segnale di "fine" viene apposto automaticamente dal computer tutte le volte che imponiamo il comando CLOSE. In altre parole il computer è in grado di capire quando un file non contiene più dati solo se è presente tale segnale: in caso contrario, quindi, la procedura rischia di diventare non più controllabile e, in ogni caso, non è possibile rintracciare correttamente i dati richiesti.

Nel caso del disco i file che vengono aperti in scrittura e non chiusi, vengono indicati, nella directory, con un asterisco e risultano esser "vuoti" (numero blocchi del file=0).

Nel caso del registratore, invece, il computer, in fase di lettura, leggerà correttamente i dati fintantochè sarà in grado di individuarli; quando troverà il nastro "vuoto", continuerà la ricerca... all'infinito senza emettere, purtroppo(!), alcuna segnalazione di errore. Figurarsi che pasticcio succede se il file non chiuso era stato scritto, in fase di registrazione, non su di un

nastro vergine ma su un nastro riciclato, contenente, cioè, altri file...

Un solo errore è possibile compiere senza che ciò pregiudichi la procedura: chiudere un file nonostante questo non sia stato aperto in precedenza. Tale possibilità consente operazioni del tipo...

CLOSE1:OPEN1,1,1,"PROVA"

...che garantiscono la possibilità, sempre e comunque, di aprire un file senza pericolo che venga emesso il messaggio "File open error".

Come scrivere i dati

Dopo aver visto come aprire (e chiudere) un file, sarebbe il caso, finalmente, di soffermarci su ciò che è possibile scrivere!

Un file aperto per scrivere possiamo considerarlo come un canale verso il quale inviare i dati più disparati. Ma, è risaputo, nel campo dei calcolatori è possibile trattare valori numerici oppure stringhe alfanumeriche. Qualunque "cosa" può essere inclusa in una delle due categorie appena definite: alla prima possiamo assegnare i numeri presenti in un estratto conto bancario, le misure di aree, pesi, pressioni, coefficienti di equazioni, eccetera; alla seconda, invece, i nomi, i cognomi, le unità di misura ed altri termini che contenessero caratteri alfabetici definiti con il termine STRINGA. Alcune "parole", però, possono esser considerate, a seconda dei casi, un numero oppure una stringa. Pensate alla data di nascita di un nostro amico oppure al numero civico del suo indirizzo: pur essendo, a tutti gli effetti, valori numerici, è possibile considerarli stringhe anche perchè, di norma, non vengono sottoposti a calcoli. Il numero 1987, quindi, potrà esser considerato una stringa, e trattato come tale, come pure 120387 può rappresentare, secondo una banale codificazione, il giorno 12 marzo 1987.

I lettori più fedeli, che ci seguono da tempo, sanno che i computer Commodore utilizzano sempre sette byte per memoriz-

zare un numero decimale, qualunque sia il suo valore: non ci soffermeremo su questo aspetto della questione, del resto già affrontato su altri numeri di CCC, che, paradossalmente, richiede sette byte sia per memorizzare "1" che "3.12345678". Ci limiteremo a sottolineare, però, che la stringa "1" è lunga un solo byte, mentre la stringa "3.12345678" ne è lunga ben 10.

Il motivo per cui teniamo a puntualizzare quanto abbiamo affermato, è dovuto al fatto che il computer, quando registra un file sequenziale, trasmette i valori sempre sotto forma di stringa, anche se questi sono valori numerici.

Supponiamo di aver aperto un file e di utilizzare il comando PRINT# (NON abbreviabile con "?") per memorizzarvi un valore numerico:

```
100 OPEN1,1,1
110 PRINT#1,100
120 CLOSE1
```

Il numero 100 non verrà trascritto con l'invio di sette byte, ma sotto forma di stringa: "100". Una successiva operazione di lettura trasformerà automaticamente, se il caso, la stringa "100" nel numero 100.

Il valore prima scritto, quindi, potrà esser letto con il seguente mini listato:

```
100 OPEN1
110 INPUT#1,A
120 PRINT A
130 CLOSE1
```

oppure con il seguente:

```
100 OPEN1
110 INPUT#1,A$
120 PRINT A$
130 CLOSE1
```

Naturalmente non è possibile il contrario. Se il programma che "scrive" è il seguente...:

```
100 OPEN1,1,1
110 PRINT#1,"PIPPO"
120 CLOSE1
```

...sarà possibile leggerlo solo con il programma che contiene INPUT#1,A$ e non INPUT#1,A. Tentando di farlo si otterrebbe, infatti, il messaggio "File data error" che comunica, appunto, l'impossibilità di assegnare ad una variabile numerica (A) una stringa alfanumerica ("PIPPO").

Altri errori che comunemente si verificano nel gestire un file sono:

• *"Not input file error". Viene visualizzato tentando di leggere un dato da un file aperto, al contrario, per registrare. Esempio:*

```
100 OPEN1,1,1
110 INPUT#1,"PIPPO"
120 CLOSE1
```

• *"Not output file error". Viene visualizzato nel caso contrario: quando si tenta di scrivere un dato in un file aperto, invece, per leggere. Esempio:*

```
100 OPEN1
110 PRINT#1,"PIPPO"
120 CLOSE1
```

• *"File too long error". Questo tipo di errore, possibile solo in lettura, è opportuno che venga spiegato in dettaglio. Supponiamo di scrivere un programma che consenta, grazie alla possibilità di concatenare più stringhe tra loro, la formazione di una stringa lunga ben 100 caratteri:*

```
100 FOR I=1TO100
110 A$=A$+"A"
120 NEXT
130 OPEN1,1,1
140 PRINT#1,A$
150 CLOSE1
```

Poichè è possibile realizzare stringhe di una lunghezza massima di 255 caratteri, l'operazione è perfettamente lecita. Se, però, tentiamo di leggere la stringa appena scritta con il solito listato...:

```
100 OPEN1
110 INPUT#1,A$
120 PRINT A$
130 CLOSE1
```

... notiamo, con costernazione, la comparsa del messaggio prima citato. Ciò è dovuto alla sintassi tipica del comando INPUT che non accetta MAI più di 80 caratteri. Se, infatti, utilizziamo INPUT da tastiera...

```
100 INPUT A$
110 PRINT A$
120 GOTO100
```

... ci rendiamo conto che non è possibile far accettare al computer un messaggio che sia lungo più di 80 caratteri. Nel caso, invece, di un dato proveniente da nastro (o disco) è come se si cercasse di "forzare" il computer ad accettare una stringa di lunghezza maggiore del consentito. In casi come questi viene attivata, inevitabilmente, la procedura di individuazione dell'errore che, nel caso di input da tastiera, non viene presa in considerazione dal momento che mai, da tastiera, la stringa può superare la lunghezza massima prefissata.

Morale: fate in modo che le stringhe inviate verso un file non superino mai gli 80 caratteri.

Come leggere, quindi, la stringa A$ lunga 100 caratteri precedentemente memorizzata? Solo con l'istruzione GET# che, prelevando un solo carattere alla volta, non ha limitazioni di alcun genere:

```
100 OPEN1
110 GET#1,A$
120 PRINT A$;
140 GOTO110
150 CLOSE1
```

Ci accorgiamo, però, che i cento caratteri vengono, sì, letti, ma il computer non sembra accorgersi che il file contiene una sola stringa. E' quindi necessario introdurre una piccola variante che consenta di accorgersi quando il file è terminato e non contiene altri dati. Il compito di individuare la fine del file è affidato alla variabile riservata ST che, di norma posta a zero, viene automaticamente modificata dal computer non appena qualcosa non va pèr il verso giusto nella gestione di un file. Apportando la seguente modifica...:

```
100 OPEN1
110 GET#1,A$
120 PRINT A$;
140 IF ST=0THEN110
150 CLOSE1
```

... non appena ST diventerà diverso da zero il programma terminerà.

Per variabile "riservata" si intende quella che può esser letta dall'utente ma non modificata, pena l'emissione di un Syntax error (provate a digitare: ST=3 o altri valori).

Un breve riepilogo

Prima di passare all'ultimo paragrafo è bene fare il punto della situazione ricordando, per ciascun comando, gli errori frmali e logici che è possibile commettere.

OPEN

Errori formali:

• *mancanza di alcuni (o tutti) i parametri necessari (Syntax Error).*
• *apertura di un file già aperto (File open error).*
• *mancata accensione della periferica interessata (Device not present error).*
• *mancanza di segnale di EOT (Device not present error).*
• *numero elevato di file aperti (Too many files)*

• *numero illegale di periferica se DN risulta minore di zero oppure 〉255 (Illegal quantity error).*
• *assenza all'interno del nome, nel caso di gestione del floppy disk, del suffisso ",R" (lettura) oppure ",W" (scrittura).*

Errori logici
• *mancanza di collegamento con il terminale RS-232 (Open2,2,...). Effetto: invio di dati che nessuno riceverà; attesa infinita di dati che non è possibile ricevere.*
• *apertura di un file che, in seguito, non verrà preso in considerazione. Effetto: occupazione inutile di canali aperti con il rischio di "Too many file error" se si supera il numero di 10 file aperti contemporaneamente.*
• *assenza di indirizzo secondario necessario per impartire particolari ordini. Effetto: apparente "disobbedienza" della periferica, tipicamente la stampante.*

CLOSE
Errori formali
• *assenza di indicazione del LF (Syntax error).*
• *valore LF elevato o negativo (Illegal quantity error).*
• *sovrabbondanza di parametri; esempio: CLOSE1,2,3,4 (Type mismatch error).*

Errori logici
• *chiusura involontaria di un file che doveva restare aperto. Effetto: perdita di dati; successivo File not open error.*
• *chiusura di un file aperto verso la stampante senza un preventivo "scarico" dei dati del buffer. Effetto: perdita attuale di alcuni dati e successivo indesiderato "recupero" del buffer nel momento in cui si aprirà nuovamente il file e si chiederà la stampa di alcuni caratteri.*

PRINT#
Errori formali
• *mancata apertura del file (File not open error).*
• *mancata indicazione del LF (Syntax error).*

• *tentativo di scrivere in un file aperto per leggere (File output data error).*

Errori logici

• *scrittura di un dato in un file piuttosto che in un altro. Effetto: ritrovare, in seguito, dati estranei ad un file e l'assenza degli stessi nell'altro file al quale erano destinati.*
• *presenza di caratteri "particolari" come virgola (,) punto e virgola (;) doppio punto (:) ritorno carrello CHR$(13). (Vedi dopo per maggiori dettagli).*
• *mancata registrazione di alcuni dati oppure dati registrati più di una volta, che interferiscano con l'eventuale "modulo" impostato nella registrazione dell'intero file.*
• *presenza del carattere speciale CHR$(34) (virgolette) che potrebbe creare problemi in fase di lettura dei dati.*

INPUT#, GET#

Errori formali

• *mancata apertura del file (File not open error).*
• *mancata indicazione di FN (Syntax error).*
• *tentativo di leggere dati in un file aperto per scrivere (File input data error).*
• *lunghezza eccessiva, nel solo caso di INPUT, della stringa da leggere (String too long error).*
• *presenza di caratteri speciali all'interno della stessa stringa (Extra ignored): solo con INPUT#.*

Errori logici

• *lettura di un dato non corrispondente a quello desiderato. Effetto: lettura del Nome invece del Cognome, e simili.*
• *scarto di un dato da considerare. Effetto: lettura del Nome e sua manipolazione, invece del Cognome, e simili.*

Un errore comune

Molti libri sulla gestione dei file contengono errori di non lieve entità che hanno gettato nella più nera disperazione numerosi utenti dei computer Commodore.

Tali errori sono dovuti, soprattutto, alla superficialità con cui l'autrice dei volumi citati gestisce i dati da memorizzare. Al momento della registrazione, infatti, una elevata disinvoltura, fortemente discutibile, non genera visualizzazioni di errori e tutto sembra che si sia svolto correttamente. I dolori vengono poi, in fase di lettura del file...

Prima di procedere è bene puntualizzare quanto segue:

• Ogni qualvolta si invia un comando di scrittura relativo, poniamo, ad una variabile, viene sempre inviato, subito dopo, ANCHE un comando di ritorno carrello (a meno di aggiungere uno dei due caratteri punto e virgola oppure virgola). Se, ad esempio, digitate sulla tastiera...:

PRINT "PIPPO"

... viene inviata, al video, la successione di caratteri: "P", "I", "P", "P", "O", CHR$(13). Ciò avviene anche se la stringa viene inviata (dopo l'opportuna apertura del file) alla stampante...

PRINT#1,"PIPPO"

... oppure alla memoria di massa (registratore o drive che sia). Se, infatti, ricorrete ad un ciclo For...Next, ottenete, in bell'ordine, l'uno sotto l'altro, i dieci nomi "PIPPO":

100 FOR I=1TO10
110 PRINT "PIPPO"
120 NEXT

Il fatto che i dieci nomi compaiano l'UNO SOTTO L'ALTRO significa, infatti, che dopo ciascuno di essi il computer appone, AUTOMATICAMENTE, un carattere di ritorno carrello CHR$(13) che, nel caso di video o stampante, produce l'effetto di spostare la testina di scrittura (o il pennello di elettroni nel caso del video) all'inizio del rigo successivo disponibile.

Se però, come già detto, inseriamo, subito dopo "PIPPO", un carattere di virgola (,) o di punto e virgola (;) impediamo al calcolatore di inserire il ritorno carrello che verrà posto, AUTOMATICAMENTE, dopo il primo PRINT relativo a.. nessuna variabile, oppure se lo inseriremo noi appositamente. In altre parole i due programmi seguenti hanno lo stesso effetto:

90 REM primo programma

```
100 FOR I=1TO10
110 PRINT"PIPPO"
120 NEXT
 90 REM secondo programma
100 FOR I=1TO10
110 PRINT"PIPPO";
115 PRINT CHR$(13);
120 NEXT
```

Si noti la maggior lunghezza del secondo listato, la necessità di inserire due caratteri di punto e virgola (pena la visualizzazione di un rigo sì ed uno no) e la presenza del CHR$(13).

Poichè l'effetto che ne risulta è IDENTICO nei due casi, risulta priva di utilità la metodologia seguita nel secondo procedimento a meno di essere masochisti.

Ma c'è un motivo molto, molto valido per non trascrivere MAI caratteri "speciali" (;,:). E' sufficiente una minima disattenzione per generare file dal contenuto inutilizzabile.

La soluzione è nelle vostre mani

Abbiamo fin qui parlato dei principali inconvenienti che possono capitare a chi si accinge a lavorare sui file.

Tutte le notizie riportate derivano in parte dai manuali ufficiali Commodore, in parte da numerose ore passate alla tastiera e in parte dallo studio degli errori logici contenuti nei libri citati.

Siamo però convinti che una maggior padronanza della gestione dei file è possibile soltanto lavorando assiduamente e creando, oppure studiando, file creati con programmi di ogni tipo simili, ad esempio, a quelli riportati in queste pagine.

La decisione, quindi, di non commentare i listati pubblicati, se non limitandosi alle note presenti nelle REM, non è dovuta a trascuratezza o, peggio, a sentimenti ostili nei confronti dei lettori che ci abbiano fin qui seguiti.

L'invito a studiarli, scriverli, testarli, modificarli e verificare, soprattutto, la correttezza di tali modifiche, va quindi inteso come un incoraggiamento, l'unico valido, allo studio di una tecnica di programmazione alla quale chiunque, prima o poi, deve approdare.

```
100 REM PROGRAMMA N.1
110 :
120 REM SCRITTURA FILE SEQUENZIALE
130 REM QUALSIASI COMPUTER COMMODODRE
140 :
150 PRINTCHR$(147)"1- NASTRO"
160 PRINT"2- DISCO"
170 GET A$:IF A$="" THEN 170
180 IF A$="1" THEN 290:REM SE NASTRO..
    .
190 IF A$="2" THEN 330:REM SE DISCO...
200 GOTO 150:REM RIFIUTA ALTRI TASTI
210 :
220 REM SINTASSI NECESSARIA:
230 REM OPEN LF,DN,SA
240 REM LF=FILE LOGICO
250 REM DN=NUMERO DI DEVICE (PERIFERIC
    A)
252 REM DN=1: NASTRO CASSETTA
254 REM DN=8: DISK DRIVE
260 REM SA=SECOND.ADDRESS (INDI.SECOND
    .)
270 :
280 REM SINTASSI NECESSARIA PER NASTRO
290 OPEN 1,1,1,"DIECI NUMERI"
300 GOTO 390
310 :
320 REM SINTASSI NECESSARIA PER DISCO
330 OPEN 1,8,1,"DIECI NUMERI,S,W"
335 REM S:SEQUENZIALE
336 REM W:WRITE (SCRIVI)
340 GOTO 390
350 :
360 REM SCRITTURA DEI DIECI NUMERI
370 REM SUL CANALE PRECEDENTEMENTE
380 REM APERTO (NASTRO O DISCO)
390 FOR I=1 TO 10:REM CICLO FOR...NEXT
400 PRINT#1,I:NEXT:REM SCRIVE NUMERI
410 CLOSE 1:END :REM TERMINE PROCEDURA
```

```
100 REM PROGRAMMA N.2
110 :
120 REM LETTURA DEL FILE SEQUENZIALE G
    ENERATO CON IL PROGRAMMA N.1
130 :
140 PRINTCHR$(147)"1- NASTRO"
```

```
150 PRINT"2- DISCO"
160 GET A$:IF A$="" THEN 160
170 IF A$="1" THEN 220:REM SE NASTRO..
    .
180 IF A$="2" THEN 260:REM SE DISCO...
190 GOTO 140:REM RIFIUTA ALTRI TASTI
200 :
210 REM SINTASSI NECESSARIA PER NASTRO
220 OPEN 5,1,0:REM NOME NON NECESSARIO
230 GOTO 340
240 :
250 REM SINTASSI NECESSARIA PER DISCO
260 OPEN 5,8,8,"DIECI NUMERI,S,R"
270 REM S=SEQUENZIALE
280 REM R=READ (=LEGGI)
290 GOTO 340
300 :
310 REM LETTURA DEI DIECI NUMERI
320 REM SUL CANALE PRECEDENTEMENTE
330 REM APERTO (NASTRO O DISCO)
340 FOR I=1 TO 10:REM CICLO FOR...NEXT
350 INPUT#5,I:REM LEGGE NUMERI
360 PRINTI:REM VISUALIZZA VALORE LETTO
370 NEXT:REM CHIUSURA CICLO
380 CLOSE 5:END :REM TERMINE PROCEDURA
```

```
100 REM PROGRAMMA N.3
110 :
120 REM LETTURA DI QUALUNQUE FILE
130 REM SEQUENZIALE DI TIPO NUMERICO
140 :
150 PRINTCHR$(147)"1- NASTRO"
160 PRINT"2- DISCO"
170 GET A$:IF A$="" THEN 170
180 IF A$="1" THEN 230:REM SE NASTRO..
    .
190 IF A$="2" THEN 270:REM SE DISCO...
200 GOTO 150:REM RIFIUTA ALTRI TASTI
210 :
220 REM SINTASSI NECESSARIA PER NASTRO
230 OPEN 8:REM SINTASSI "MINIMA"
240 GOTO 330
250 :
260 REM SINTASSI NECESSARIA PER DISCO
270 OPEN 8,8,8,"DIECI NUMERI,S,R"
```

```
280 GOTO 330
290 :
300 REM LETTURA DEI VALORI NUMERICI
310 REM SUL CANALE PRECEDENTEMENTE
320 REM APERTO (NASTRO O DISCO)
330 INPUT#8,I:REM LEGGE NUMERO
340 PRINTI:REM VISUALIZZA VALORE LETTO
350 IF ST>0 THEN 370:REM ESAME ST
360 GOTO 330:REM RICOMINCIA
370 IF A$="2" THEN 460:REM SE DISCO
380 PRINT:PRINT"ST VALE:"ST;
390 IF (ST  AND 64)=64 THEN PRINT"FINE
     FILE"
400 IF (ST  AND 32)=32 THEN PRINT"CHEC
    KSUM ERROR (CASSETTA)"
410 IF (ST  AND 16)=16 THEN PRINT"ERRO
    RE DI LETTURA (CASSETTA)"
420 IF (ST  AND 8)=8 THEN PRINT"BLOCCO
     LUNGO (CASSETTA)"
430 IF (ST  AND 4)=4 THEN PRINT"BLOCCO
     CORTO (CASSETTA)"
440 IF (ST  AND 2)=2 THEN PRINT"ERRORE
     DI TEMPORIZZAZIONE (CASSETTA)"
450 GOTO 500:REM ESCLUDE LETTURA CANAL
    E DI ERRORE IDONEO PER IL SOLO DIS
    CO
460 OPEN 15,8,15:REM APERTURA CANALE E
    RRORE (SOLO PER DISCO)
470 INPUT#15,A,A$,B,C:REM LEGGE DA DIS
    CO I VARI CODICI ERRORE
480 PRINTA;A$;B;C:REM VISUALIZZA I MES
    SAGGI LETTI PRECEDENTEMENTE
490 CLOSE 15:REM CHIUDE CANALE ERRORE
500 CLOSE 8:END :REM TERMINE INTERA PR
    OCEDURA SIA PER DISCO CHE PER NAST
    RO
```

```
100 REM PROGRAMMA N.4
110 :
120 REM SCRITTURA FILE SEQUENZIALE
130 REM FORMATO DA VALORI NUMERICI
140 REM E STRINGHE ALFANUMERICHE
150 :
160 PRINTCHR$(147)"1- NASTRO"
170 PRINT"2- DISCO"
180 GET A$:IF A$="" THEN 180
```

```
190 IF A$="1" THEN 240:REM SE NASTRO..
    .
200 IF A$="2" THEN 280:REM SE DISCO...
210 GOTO 160:REM RIFIUTA ALTRI TASTI
220 :
230 REM SCRIVE NUMERI E PAROLE
240 OPEN 1,1,2,"NUMERI PAROLE":REM PRE
    STARE ATTENZIONE AL SA=2
250 GOTO 340
260 :
270 REM SINTASSI NECESSARIA PER DISCO
280 OPEN 1,8,1,"NUMERI PAROLE,S,W"
290 GOTO 340
300 :
310 REM SCRITTURA DEI VENTI DATI
320 REM SUL CANALE PRECEDENTEMENTE
330 REM APERTO (NASTRO O DISCO)
340 FOR I=1 TO 10:REM APERTURA DEL PRI
    MO CICLO FOR...NEXT RELATIVO AI NU
    MERI
350 PRINT#1,I:NEXT:REM SCRIVE NUMERI
360 FOR I=1 TO 10:REM APERTURA DEL SEC
    ONDO CICLO FOR...NEXT RELATIVO ALL
    E STRINGHE
370 PRINT#1,"PAROLA":NEXT:REM SCRIVE P
    AROLE
380 CLOSE 1:END :REM TERMINE PROCEDURA
```

```
100 REM PROGRAMMA N.5
110 :
120 REM LETTURA DI QUALUNQUE FILE SEQU
    ENZIALE
130 REM NUMERICO OPPURE STRINGA MEDIAN
    TE IL RICORSO ALL'ISTRUZIONE GET#
140 :
150 PRINTCHR$(147)"1- NASTRO"
160 PRINT"2- DISCO"
170 GET A$:IF A$="" THEN 170
180 IF A$="1" THEN 230:REM SE NASTRO..
    .
190 IF A$="2" THEN 270:REM SE DISCO...
200 GOTO 150:REM RIFIUTA ALTRI TASTI
210 :
220 REM SINTASSI NECESSARIA PER NASTRO
230 OPEN 8:REM SINTASSI "MINIMA"
240 GOTO 360
```

```
250 :
260 REM SINTASSI NECESSARIA PER DISCO
270 INPUT "NOME FILE";NF$:IF NF$="" TH
    EN 270
280 IF LEN(NF$)>16 THEN 270
290 NF$=NF$+",S,R"
300 OPEN 8,8,8,NF$
310 GOTO 360
320 :
330 REM LETTURA DEI SINGOLI CARATTERI
340 REM SUL CANALE PRECEDENTEMENTE
350 REM APERTO (NASTRO O DISCO)
360 GET #8,X$:REM LEGGE SINGOLO CARATT
    ERE
370 PRINTX$;:REM E LO VISUALIZZA (ATTE
    NZIONE AL PUNTO E VIRGOLA FINALE)
380 IF ST>0 THEN 400:REM ESAME ST
390 GOTO 360:REM RICOMINCIA
400 IF A$="2" THEN 430:REM SE DISCO
410 PRINT:PRINT"ST VALE:"ST;
420 GOTO 470
430 OPEN 15,8,15
440 INPUT#15,A,A$,B,C
450 PRINT:PRINTA;A$;B;C
460 CLOSE 15
470 CLOSE 8:END :REM TERMINE INTERA PR
    OCEDURA SIA PER DISCO CHE PER NAST
    RO
```

```
100 REM PROGRAMMA N.6
110 :
120 REM SCRITTURA DI EVENTUALI CARATTE
    RI NULLI ED ESAME SUCCESSIVO
130 REM (MEDIANTE IL PROGRAMMA N.5) DE
    I MALFUNZIONAMENTI RISCONTRABILI
140 :
150 PRINTCHR$(147)"1- NASTRO"
160 PRINT"2- DISCO"
170 GET A$:IF A$="" THEN 170
180 IF A$="1" THEN 220:REM SE NASTRO..
    .
190 IF A$="2" THEN 260:REM SE DISCO...
200 GOTO 150:REM RIFIUTA ALTRI TASTI
210 REM NASTRO
220 OPEN 1,1,1,"CARATTERI NULLI"
230 GOTO 320
```

```
250 REM DISCO
260 OPEN 1,8,8,"CARATTERI NULLI,S,W"
270 GOTO 320
290 REM SCRITTURA DI STRINGHE ALFAN.
300 REM SUL CANALE PRECEDENTEMENTE
310 REM APERTO (NASTRO O DISCO)
320 X$="":REM ANNULLAMENTO "PREVENTIVO
    " DELLA STRINGA JOLLY ALFANUMERICA
330 INPUT "DIGITA LA STRINGA";X$:REM D
    IGITAZIONE CARATTERI
340 PRINT#1,X$:REM SCRITTURA SU FILE
350 IF X$<>"*" THEN 320:REM CONTINUAZI
    ONE DIGITAZIONE SE NON CARATTERE "
    CODICE"
360 CLOSE 1:END :REM TERMINE PROCEDURA
```

```
100 REM PROGRAMMA N.7
120 REM SCRITTURA DI "OGGETTI" SEPARAT
    I DA VIRGOLA E PUNTO E VIRGOLA
130 REM PER SUCCESSIVA LETTURA
140 REM (MEDIANTE IL PROGRAMMA N.8) DE
    I MALFUNZIONAMENTI RISCONTRABILI
160 PRINTCHR$(147)"1- NASTRO"
170 PRINT"2- DISCO"
180 GET A$:IF A$="" THEN 180
190 IF A$="1" THEN 230:REM SE NASTRO..
    .
200 IF A$="2" THEN 270:REM SE DISCO...
210 GOTO 160:REM RIFIUTA ALTRI TASTI
220 REM NASTRO
230 OPEN 1,1,1,"VIRGOLA"
240 GOTO 300
260 REM DISCO
270 OPEN 1,8,8,"VIRGOLA,S,W"
280 GOTO 300
300 A=10:B=20:C=3.14:REM IMPOSTAZIONE
    DI ALCUNI VALORI NUMERICI
310 A$="PRIMO":B$="SECONDO":C$="TERZO"
    :REM STRINGHE ALFANUMERICHE
320 PRINT#1,A,B,C:REM SEPARAZIONE VIRG
    OLA TRA VALORI NUMERICI
330 PRINT#1,A;B;C:REM SEPARAZIONE PUNT
    O E VIRGOLA TRA VALORI NUMERICI
340 PRINT#1,A$,B$,C$:REM SEPARAZIONE V
    IRGOLA TRA STRINGHE
350 PRINT#1,A$;B$;C$:REM SEPARAZIONE P
```

```
   UNTO E VIRGOLA TRA STRINGHE
360 CLOSE 1:END :REM TERMINE PROCEDURA
```

```
100 REM PROGRAMMA N.8
120 REM LETTURA DI QUALUNQUE FILE SEQU
    ENZIALE
130 REM NUMERICO OPPURE STRINGA MEDIAN
    TE IL RICORSO ALL'ISTRUZIONE INPUT #
150 PRINTCHR$(147)"1- NASTRO"
160 PRINT"2- DISCO"
170 GET A$:IF A$="" THEN 170
180 IF A$="1" THEN 220:REM SE NASTRO..
190 IF A$="2" THEN 260:REM SE DISCO...
200 GOTO 150:REM RIFIUTA ALTRI TASTI
220 OPEN 8:REM NASTRO
230 GOTO 350
250 REM DISCO
260 INPUT "NOME FILE";NF$:IF NF$="" TH
    EN 260
270 IF LEN(NF$)>16 THEN 260
280 NF$=NF$+",S,R"
290 OPEN 8,8,8,NF$
300 GOTO 350
320 REM LETTURA DELLE SINGOLE STRINGHE
330 REM SUL CANALE PRECEDENTEMENTE
340 REM APERTO (NASTRO O DISCO)
350 INPUT#8,X$:REM LEGGE STRINGA
360 PRINTX$:REM LA VISUALIZZA...
370 PRINT"LUNGHEZZA:"CHR$(18)LEN(X$):R
    EM ...INSIEME ALLA SUA LUNGHEZZA..
380 PRINT"VALORE:"CHR$(18)VAL(X$):REM
    ...E AL SUO VALORE
390 PRINT
400 IF ST>0 THEN 420:REM ESAME ST
410 GOTO 350:REM RICOMINCIA
420 IF A$="2" THEN 450:REM SE DISCO
430 PRINT:PRINT"ST VALE:"ST;
440 GOTO 490
450 OPEN 15,8,15
460 INPUT#15,A,A$,B,C
470 PRINT:PRINTA;A$;B;C
480 CLOSE 15
490 CLOSE 8:END :REM TERMINE INTERA PR
    OCEDURA SIA PER DISCO CHE PER NAST
    RO
```